



SEVENTH FRAMEWORK PROGRAMME INFORMATION AND COMMUNICATION TECHNOLOGIES

Project:

Accessibility Assessment Simulation Environment for New
Applications Design and Development
(ACCESSIBLE, Grant Agreement No. 224145)



Deliverable number and title:

D 5.5b – ACCESSIBLE prototype (final version)

Lead beneficiary:	CERTH / ITI
WP. no, title and activity type	WP5 Core Development
Contributing Task (s)	T5.5 Integration of ACCESSIBLE Modules and Tools
Dissemination level	Confidential and Public version
Delivery date	Month 42
Status	Final Draft
Dissemination level	Co (Confidential) & Public Part
File name and size	"D5.5b_ACCESSIBLE_WP5_CERTH-FD-R-v1.0.pdf", 44946 Kb

Authors List

Leading Authors (<i>Editors</i>)				
	<i>Surname</i>	<i>Initials</i>	<i>Beneficiary Name (Short Name)</i>	<i>Contact email</i>
	Tzovaras	D.	CERTH/ITI	Dimitrios.Tzovaras@iti.gr
	Votis	K.	CERTH/ITI	kvotis@iti.gr
Co-authors (<i>In alphabetic order</i>)				
#	<i>Surname</i>	<i>Initials</i>	<i>Beneficiary Name (Short Name)</i>	<i>Contact email</i>
1.	Kaklanis	N.	CERTH/ITI	nkak@iti.gr
2.	Giakoumis	D.	CERTH/ITI	dgiakoumis@iti.gr
3.	Tsakiris	A.	CERTH/ITI	atsakir@iti.gr
4.	Carrico	L	FFCUL	luis.lmc@di.fc.ul.pt
5.	Bandeira	R.	FFCUL	rbandeira@lasige.di.fc.ul.pt
6.	Carvalho	M.	SOLINET	M.Carvalho@SOLINET.com
7.	Partarakis	N.	FORTH	partarak@ics.forth.gr
8.	Zdenek	M.	CVUT	xmikovec@fel.cvut.cz

Peer Reviewers List

#	<i>Surname</i>	<i>Initials</i>	<i>Contact email</i>
1.	Panou	MP	mpanou@certh.gr
2.	Van Isacker	KVI	projects@marie-curie-bg.org

History Table

Date	Comments
30/11/2011	First draft version
25/01/2012	Second version integrating input from partners CVUT and CERTH
24/02/2012	Third version integrating input from partners SOLINET, CERTH, FORTH and FFCUL
28/02/2012	Overall review of the document and minor changes
29/02/2012	Final version and sent for peer review

Executive Summary

This deliverable (software and report) outlines the description of the final implemented ACCESSIBLE prototypes, which include the Web Accessibility assessment module (integrating also the ARIA assessment module), the Web Services Accessibility assessment module (supporting also the WADL services), the Mobile Web applications assessment module, and the Description Languages assessment tool which has been integrated in the SAFIRE IDE. The aforementioned assessment modules are successfully integrated to the ACCESSIBLE portal, a Unified Interface that is available to all users, for all the facilities developed within the premises of the ACCESSIBLE project. Moreover, the Mobile Impairment simulator (MIS) tool and the Disability Impairment Approximation Simulation (DIAS) are being described along with the changes since the updated version.

The purpose of this document is to describe the functionality of the aforementioned tools, the implemented assessment tests, the benefits that they provide to users and to provide a tutorial on how to use them through the portal, through appropriate IDEs and/or as standalone modules. This document is an updated version of the deliverable D5.5a (first version) delivered on month 32, which is an output of work package 5 and more specifically the task T5.5, thus it describes the main functionalities of the ACCESSIBLE components.

More technical information concerning the architecture, the requirements of the tools and the related technologies can be extracted by the deliverables D5.2 “Assessment Simulation Module”, D5.3 “developer designer aid module” and D5.4 “EARL based reporting module”.

Table of contents

Authors List	3
Peer Reviewers List	3
History Table	4
Executive Summary	5
Table of contents	6
List of figures	9
List of tables	12
List of abbreviations and acronyms	13
1. Introduction	14
1.1. Scope	14
2. Introduction to the ACCESSIBLE portal and Standalone User Interface tool....	16
2.1. Design Process	16
2.2. Architecture of the ACCESSIBLE Portal	17
2.3. Overview of the Portal	19
2.4. Overview of the Standalone User Interface Tool	20
3. Web Accessibility Assessment Tool	22
3.1. Web Accessibility Assessment Tool Implemented Tests	24
3.2 Description of Web Accessibility Assessment Tool functionalities (through the ACCESSIBLE Portal)	31
3.3 Web Accessibility Assessment (using the Standalone Tool)	44
3.4 Changes in current version	50
4. Introduction to the Web Services Assessment module	54
4.1. Web Services Accessibility Assessment (using the ACCESSIBLE Portal)	57
4.2. Web Services Accessibility Assessment (using the standalone tool)	70
4.3. Changes in current version	75
5. Mobile Web Accessibility Assessment module	76
5.1. Mobile Web Accessibility Assessment (using the tool from the ACCESSIBLE portal) 77	
5.2. Changes in current version	90
6. Description Language Assessment module	91
6.1. Short description of Description Languages Assessment Module	91
6.2. Technologies related to Description Languages Assessment Module	92
6.2.1. Specification and Description Language (SDL) and relevant technologies 92	
SAFIRE IDE Integration	92
6.3. Technologies related to the implementation of the Description Languages Assessment Module	96
6.4. Usage Scenarios for Description Languages Assessment Module (Standalone) 99	
6.5. Usage Scenarios for Description Languages Assessment Module (ACCESSIBLE Portal)	107
6.6. Changes in current version	112
7. Introduction to the Disability Impairment Approximation Simulator (DIAS) ..	113
7.1. Impairments and symptoms covered by DIAS	114
7.2. Implementation of simulations	117
7.3. Personas	122
7.4. Incorporation of Magnifier and Screen reader within simulations	122
7.5. Disability Impairment Approximation Simulation (standalone version)	124

7.6.	Disability Impairment Approximation Simulation (NetBeans plugins)	128
7.7.	Changes in current version.....	132
8.	Introduction to Mobile Impairment Simulator (MIS Tool)	133
9.	Conclusions.....	139
10.	Annex.....	140
10.1.	ACCESSIBLE Ideal Operations of the Web Services Assessment module 140	
10.2.	Controls for the Supported Impairments in DIAS	142
	References.....	145

List of figures

Figure 1: Design prototype of the ACCESSIBLE portal and standalone tool.....	17
Figure 2: Extending the ACCESSIBLE architecture with the building blocks of the user interface	18
Figure 3: Main view of the ACCESSIBLE portal	19
Figure 4: Main view of the ACCESSIBLE portal, after the login.....	20
Figure 5: Start-up screen	21
Figure 6: WaaT architecture overview	22
Figure 7: Web Accessibility Inspection Projects	31
Figure 8: Add a new project.....	31
Figure 9: Details concerning a new project	33
Figure 10: Five options for the assessment of a Web application	34
Figure 11: Evaluation against Personas, step 1.....	34
Figure 12: Evaluation against Personas, step 2.....	35
Figure 13: Evaluation against Personas, step 3.....	36
Figure 14: Report containing only errors and warnings	36
Figure 15: Report using the EARL format	37
Figure 16: Report in a human-readable EARL format (PDF document).....	38
Figure 17: Report in PDF format	38
Figure 18: Evaluation against Ontology, step 1	39
Figure 19: Evaluation against Ontology, step 2.....	40
Figure 20: Evaluation against Ontology, step 3.....	40
Figure 21: Quick evaluation against Ontology, step 1.....	41
Figure 22: Evaluation against specific guidelines, step 1	42
Figure 23: Evaluation against specific guidelines, step 2.....	42
Figure 24: Evaluation against specific guidelines, step 3.....	43
Figure 25: Quick evaluation against specific guidelines, step 1	43
Figure 26: Create new project.....	44
Figure 27: Open ACCESSIBLE project	44
Figure 28: Add new item (file from local file system)	45
Figure 29: Add new item (web-page from URL)	45
Figure 30: Source view of HTML document.....	46
Figure 31: Preview Web Page using one of the embedded browsers (Internet Explorer or Firefox)	46
Figure 32: Selection of assessment criteria through the project properties form	47
Figure 33: Web Page Assessment results: Full assessment output.....	48
Figure 34: Web Page Assessment results: Errors List	48
Figure 35: Web Page Assessment results: Warnings List	49
Figure 36: Web Page Assessment results (PDF report).....	49
Figure 37: Web Page Assessment results (EARL report in PDF format).....	50
Figure 38: WAI-ARIA ontology instances	51
Figure 39: EARL report in PDF format, containing information about Personas	52
Figure 40: Typical Web Service utilization chain	54
Figure 41: WS Accessibility Assessment Tool (WSaaT) architecture	55
Figure 42: Web Services Assessment Projects	58
Figure 43: Web Services Assessment Projects	58
Figure 44: Three options for the evaluation of the Web Services	59

Figure 45: Evaluation against Personas, step 1.....	60
Figure 46: Evaluation against Personas, step 2.....	61
Figure 47: Evaluation against Personas, step 3.....	61
Figure 48: Full report	62
Figure 49: EARL report (xml format)	63
Figure 50: EARL report (pdf format)	63
Figure 51: Evaluation against Personas, step 5.....	64
Figure 52: Evaluation against Personas, step 6.....	65
Figure 53: Evaluation against Personas, step 7.....	65
Figure 54: Evaluation against ontology, step 1.....	66
Figure 55: Evaluation against ontology, step 2.....	67
Figure 56: Evaluation against ontology, step 3.....	67
Figure 57: Evaluation against ontology, step 4.....	68
Figure 58: Evaluation against ontology, step 5.....	68
Figure 59: Evaluation against ontology, step 6.....	69
Figure 60: Evaluation against ontology, step 7.....	69
Figure 61: Evaluation against specific guidelines, step 1	70
Figure 62: WS accessibility assessment Step 1: Parsing of service definition file.....	71
Figure 63: WS accessibility assessment Step 1: The Service Definition Editor.....	71
Figure 64: Manual definition of a WS operation input/output elements	72
Figure 65: WS accessibility assessment Step 2: Automatic assessment of initial subset of guidelines.....	72
Figure 66: WS accessibility assessment Step 3: Selection of operation type, according to the accessible ideal operations	73
Figure 67: WS accessibility assessment Step 3: Alignment of operation's inputs/outputs to respective ones of the ideal operation	74
Figure 68: WS accessibility assessment Step 4: Assessment of final set of guidelines	75
Figure 69: Example of MWAAT.....	76
Figure 70: Main view of the ACCESSIBLE portal (Mobile Web Applications Assessment Projects)	77
Figure 71: Selection of project to be assessed	77
Figure 72: Four options for the evaluation of the Web Services	78
Figure 73: Evaluation against Personas, step 1.....	79
Figure 74: Evaluation against Personas, step 2.....	80
Figure 75: Evaluation against Personas, step 3.....	81
Figure 76: Full report for the MWBP checkpoints	82
Figure 77: Evaluation against ontology, step 1.....	83
Figure 78: Evaluation against ontology, step 2.....	84
Figure 79: Evaluation against ontology, step 3.....	85
Figure 80: Evaluation against specific guidelines, MobileOK guidelines.....	86
Figure 81: Evaluation against specific guidelines, WCAG2.0 guidelines.....	87
Figure 82: Evaluation against specific guidelines, step 2	88
Figure 83: Evaluation against specific guidelines, step 3	89
Figure 84: Quick evaluation against ontology, step 1.....	90
Figure 85: System Architecture of the Accessibility Assessment Module for SDL Systems.	91
Figure 86: SAFIRE Tool Chain	93
Figure 87: SAFIRE Designer.....	94
Figure 88: SAFIRE validation diagram	95

Figure 89: SAFIRE Campaigner.....	95
Figure 90: SAFIRE Animator.....	96
Figure 91: Class Diagram of the Description Languages Assesment Module	97
Figure 92: SAFIRE “Product Browser” screen.	99
Figure 93: Selection of archive to be opened.....	99
Figure 94: Browse for folder of the archive to be opened.	100
Figure 95: SAFIRE Organiser archive tree structure.....	100
Figure 96: Launching the Description Languages Accessibility Assessment Module.	101
Figure 97: Selection of SDL application to be evaluated	101
Figure 98: Manual selection of approaches	102
Figure 99: Results	103
Figure 100: Results using the EARL tool.....	103
Figure 101: Selection of approaches based on the ACCESSIBLE harmonized methodology	104
Figure 102: Selection of impairment	104
Figure 103: Selection of disabilities	105
Figure 104: Listing of Guidelines/Approaches applying for the selected impairment / disabilities.	105
Figure 105: Selection of Approaches.....	106
Figure 106: Selection of the accessibility assessment method to be used.	107
Figure 107: Selection of personas.....	107
Figure 108: Selection of Evaluation Approaches.	108
Figure 109: Accessibility Assessment Results.	108
Figure 110: Selection of Checkpoints for each defined Guideline.	109
Figure 111: Selection of Disabilities.....	110
Figure 112: Selection of Functional Limitations.	110
Figure 113: Selection of Impairments.....	111
Figure 114: Selection of evaluation categories for quick evaluation.....	112
Figure 115: Simulation of visual impairments over the UI shown in Fig. (a).....	118
Figure 116: Filters implemented for various categories of sensorineural hearing loss: (a) mild noise induced hearing loss, (b) moderate-severe noise induced hearing loss, (c) mild age -related hearing loss, (d) moderate age -related hearing loss	120
Figure 117: Instance of Parkinson simulation over a Java application.....	121
Figure 118: Instance of dyslexia simulation over a Java application	122
Figure 119: Magnifier applied over night blindness simulation.....	123
Figure 120: DIAS standalone, main view	124
Figure 121: The DIAS simulation environment	125
Figure 122: Detailed Settings for the simulation of parkinson’s disease.....	126
Figure 123: Help tab for Tritanopia	126
Figure 124: Simulation of Achromatopsia using Magnifier	127
Figure 125: DIAS standalone, initial state.....	127
Figure 126: Simulation of protanopia over a Java-based mobile application with DIAS	128
Figure 127: Toolbar of NetBeans IDE with the DIAS run main (marked in red), web preview (blue) and preview design (green) plugins installed	128
Figure 128: Simulation of glaucoma over a Swing form through the DIAS preview design plugin.....	129
Figure 129: Recommendations provided through the NetBeans IDE preview design plugin, in respect of dyslexia simulation	130

Figure 130: Simulation of a Java application with a persona suffering from blurred vision (Hyperopia) through the DIAS run main plugin	131
Figure 131: Simulation of a web page (www.google.com) with retinitis pigmentosa impairment through the DIAS web preview plugin.....	132
Figure 132: Reflection on the display and occlusion of the display with finger	134
Figure 133: Screenshot of Filter settings window with marked sections	135
Figure 134: Screenshot of MIS tool with blur effect placed over the Android emulator	136
Figure 135: Screenshot of simulated effects - reflection of fluorescent lamp and finger occlusion	137

List of tables

Table 1: Implemented tests in WaaT following accurately the test procedure defined by the guidelines of WCAG 2.0.....	24
Table 2: Implemented tests in WaaT including also some assumptions on the test procedure defined by the WCAG 2.0 guidelines	27
Table 3: Tests implemented in the WaaT according to the WAI-ARIA specifications	30
Table 4: HTTP RDF/XML report - Example	53
Table 5: The Ideal Operations defined within the Accessible Ontology, utilized by the WSaaT.....	57
Table 6: Indicative impairment parameters derived from the relevant literature, which were followed for the implementation of our simulation filters.	116
Table 7: Description of the simulation controls.....	144
Table 8: Usage of simulation controls	144

List of abbreviations and acronyms

(in alphabetic order)

API	Application Programming Interface
CSS	Cascading Style Sheets
EARL	Evaluation and Report Language
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICF	International Classification of Functioning, Disability and Health
IDE	Integrated development environment
PDF	Printable Document Format
SDL	Specification and Description Language
SOAP	Simple Object Access Protocol
URL	Uniform Resource Locator
WADL	Web Application Description Language
WAI	Web Accessibility Initiative
WCAG 2.0	Web Content Accessibility Guidelines 2.0
ARIA	Accessible Rich Internet Applications
WSDL	Web Services Description Language
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
MIS	Mobile Impairment Simulator
DIAS	Disability Approximation Simulator
WaaT	Web accessibility assessment Tool
HCI	Human Computer Interaction
UI	User Interface

1. Introduction

1.1. Scope

The basic target of this deliverable is to fully describe all the functionalities of the implemented ACCESSIBLE Prototypes (final version), namely all the ACCESSIBLE assessment modules that are integrated in the ACCESSIBLE portal as well as the impairment simulation modules DIAS and MIS. The technical details concerning the assessment and DIAS simulation module are described thoroughly in the deliverables D5.2 “Assessment Simulation Module” and D5.3 “developer and designer aid module”, whereas in this deliverable a general description of the ACCESSIBLE assessment tools, the portal, as well as the simulation tools is provided.

The integrated ACCESSIBLE assessment modules are the following:

- The Web Accessibility Assessment module, which supports the WCAG2.0 standard, and the newly integrated ARIA 1.0 standard for the accessibility assessment of Web applications
- The Web Services Assessment module, for the accessibility assessment of WSDL and WADL Web Services
- The Mobile Web assessment module for the accessibility assessment of Mobile Web applications
- The Description Languages assessment module for the accessibility assessment of Description Languages

Concerning simulation tools, a stable standalone version of the Disability Impairment Approximation Simulator (DIAS) and a NetBeans plugin version are described, while finally the new implemented MIS tool is being presented in this deliverable.

Also, this deliverable describes the changes that have been performed to the latest versions of the assessment and simulation modules, which derive either from the comments of the latest reviews (2nd and 3rd review) or from the overall comments received from the ACCESSIBLE pilot phases 2 and 3.

This deliverable is organised in the following sections:

- Section 2 gives an overview about the ACCESSIBLE interfaces (portal, standalone), the main idea behind the implementation of the portal as well as the provided basic functionalities.
- Section 3 presents thoroughly the usage scenario concerning the Web Accessibility Assessment module, through the ACCESSIBLE portal and the standalone interface. Also the modifications and the changes provided to the previous version of the tool (presented in the deliverable D5.5a) are also provided.
- Section 4 presents thoroughly the usage scenario concerning the Web Services Accessibility Assessment module.
- Section 5 gives an overview of the usage scenario concerning the Mobile Web Accessibility Assessment module.

- Section 6 presents the usage scenario concerning the Description Languages Assessment module.
- Section 7 gives an overview of the Disability Impairment Approximation Simulator (DIAS), the implemented simulations and the supported impairments, usage scenarios of both versions (standalone version and NetBeans plugin version), as well as the changes to both versions.
- Section 8 presents the new developed Mobile Impairment simulator (MIS Tool)
- Finally, in section 9 the conclusions are summarized and in section 10 an Annex with the ideal operations that are needed for the Web services assessment tool and the DIAS adopted controls are provided.

2. Introduction to the ACCESSIBLE portal and Standalone User Interface tool

The main goals for the development of the ACCESSIBLE portal were to create a User Interface that would offer an interactive portal giving access to the ACCESSIBLE assessment facilities via the Web. By these means, a Unified Interface would be available to all users, for all the facilities developed within the premises of ACCESSIBLE project.

The portal has integrated the Web Accessibility Assessment module, the Mobile Accessibility Assessment module, the Web Services Accessibility Assessment module and the Description Languages Assessment module. Moreover, the reporting functionality was also integrated to the ACCESSIBLE portal, thus providing to users reports in PDF version as well as in EARL version.

The innovation that was proposed by the ACCESSIBLE portal is that it supports multitude of standards and applications domains and there is also no need for accessing alternative applications for different application contexts. Moreover, the ACCESSIBLE portal increases the awareness regarding the need for accessibility as introduced by the ACCESSIBLE project, in different applications domains (such as Web Services, Description Languages etc.).

On the other hand the standalone tool brings the same facilities offered by the portal through a tool that can be downloaded and installed in a local computer. Such a tool provides a more versatile interface suitable for developers and allows full control of developer's evaluation projects. At the same the interface of the standalone tool resembles the ones offered by modern IDEs and thus provides new functionality using known to developers metaphors.

2.1. *Design Process*

A user centred approach has been followed for the design and implementation of the ACCESSIBLE User Interface Portal and the Stand Alone version (Figure 1). More specifically during this process the appropriate end user requirements were gathered and analysed. These requirements were categorised into:

- Generic functional Requirements (functionality that should be integrated to the tools)
- Performance requirements (requirements in terms of time needed for performing operations)
- Operational requirements (additional functionality to support system qualities such as usability and accessibility)
- Reliability requirements (requirements for producing a reliable system)
- Maintainability & Interoperability requirements (requirements for the system in order to be easily maintained and to be able to interoperate with other systems)

A detailed description of the collected requirements is presented to deliverable D5.1 "User Interface Portal & User Assistant Agent" section 3. The requirements stemming from the user requirements analysis process (D2.2a and D2.2b deliverables) were used to produce UI prototypes in an iterative process (involving the production and

evaluation of prototypes by end users). An example of such a prototype for the portal and standalone tool is presented in Figure 1 while the full design process is presented in deliverable D5.1 “User Interface Portal & User Assistant Agent” section 4.



Figure 1: Design prototype of the ACCESSIBLE portal and standalone tool

2.2. Architecture of the ACCESSIBLE Portal

The user interface portal extends the architecture of ACCESSIBLE for allowing the UI layer of the project (portal and standalone tool) to communicate with the lower levels of the architecture as presented in Figure 2.

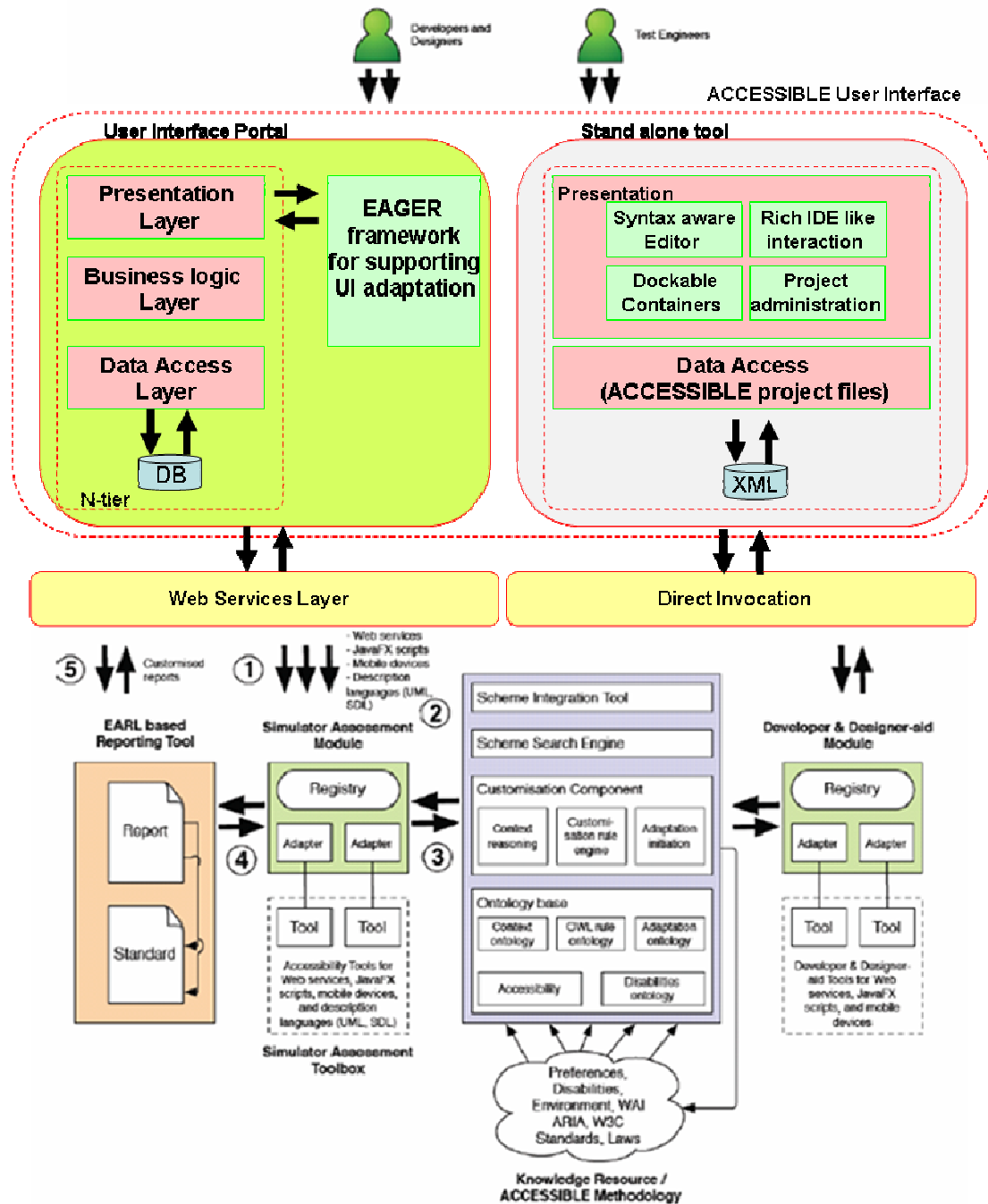


Figure 2: Extending the ACCESSIBLE architecture with the building blocks of the user interface

As presented in this figure, the user interface of ACCESSIBLE takes two distinct instantiations according to the means of access. When access is carried out through the web the user interface portal is used while a standalone application carries out the tasks of making the infrastructure available offline. A detailed description of this extended architecture is presented in deliverable D5.1 “User Interface Portal & User Assistant Agent” section 5.

2.3. Overview of the Portal

The main view of the ACCESSIBLE portal is depicted in Figure 3 and is available in the following URL <http://hci-web9.ics.forth.gr/ACCESSIBLE/central.aspx>.

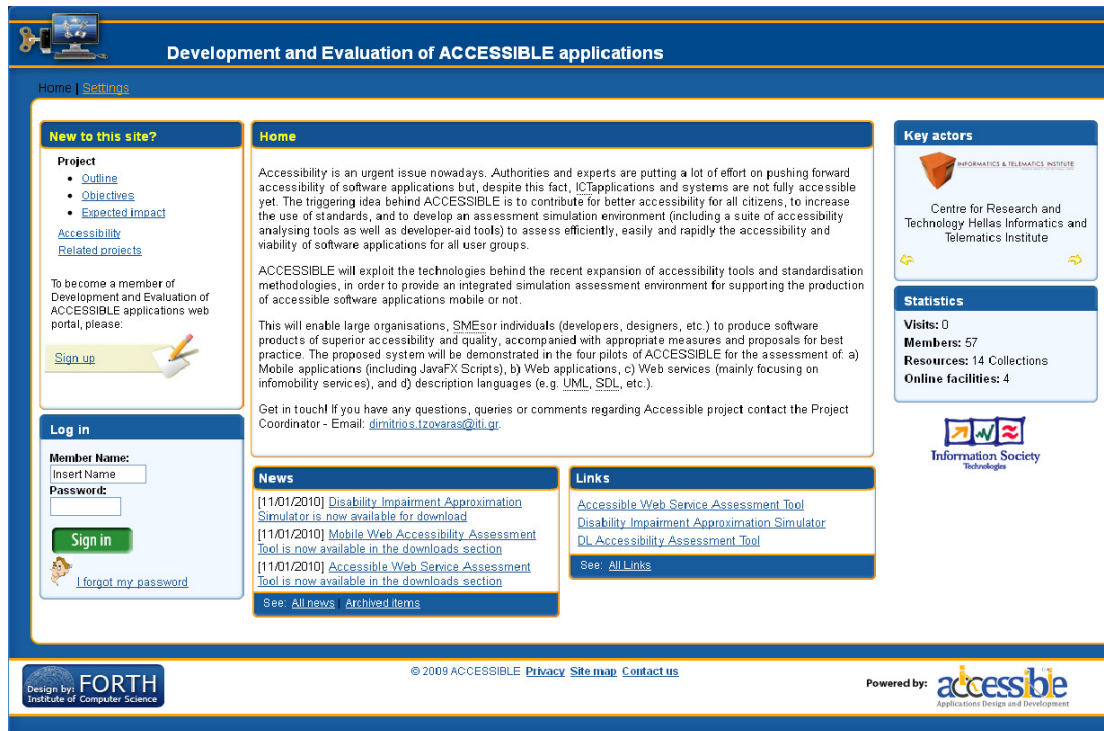


Figure 3: Main view of the ACCESSIBLE portal

The user can sign up to the ACCESSIBLE portal in order to use all the offered functionalities. Then, using the *Member Name* and the *Password*, he/she can login to the portal and the main view is presented in Figure 4.

As it is obvious, the user can access the standalone tools as well as the online assessment modules. The standalone tools are described thoroughly, along with a manual, in the deliverable D5.2 “Assessment Simulation Module” and at the same time a detailed description of the portal and its user manual is presented in D5.1 “User Interface Portal & User Assistant Agent” section 7.1. This logged in page also provides a number of quick assessment widgets. These widgets allow users to quickly start an evaluation session by simple inserting the URL of the web-page to be assessed and therefore skipping the project creation process.

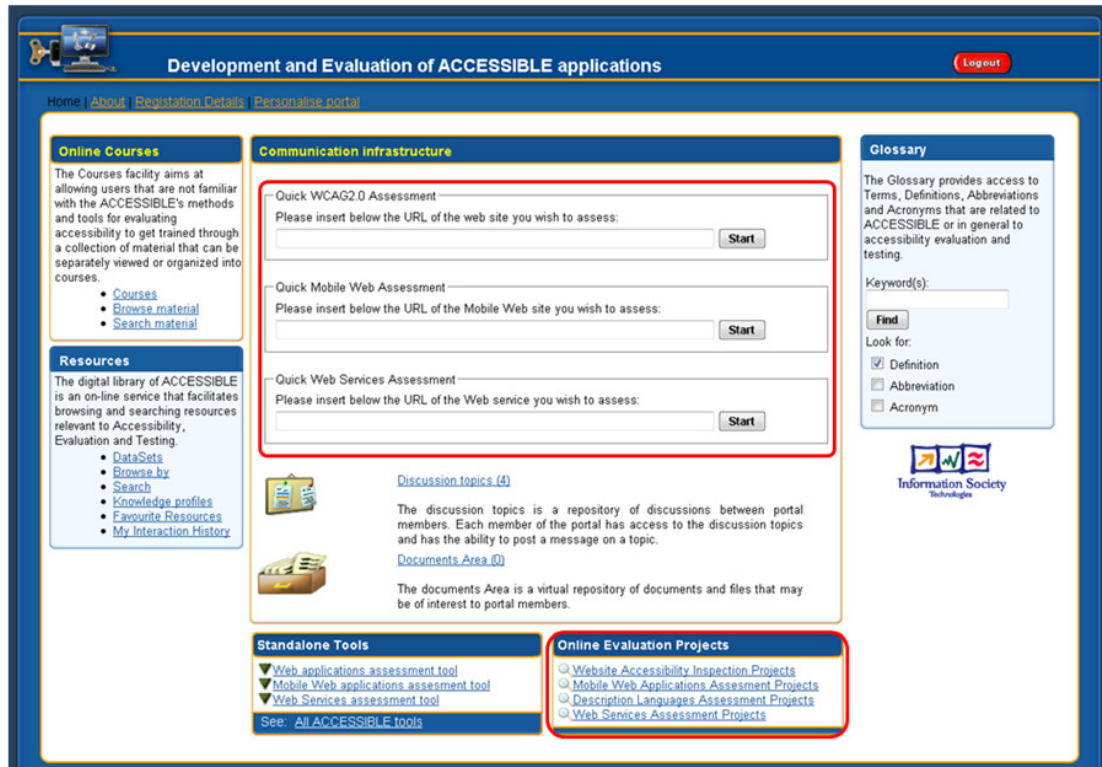


Figure 4: Main view of the ACCESSIBLE portal, after the login

The typical steps involved in performing an evaluation through the portal are

- Create an assessment project
- Add applications to be assessed
- Edit Project evaluation criteria
- Perform the assessment
- Access assessment results and fix the errors offline (with the help of an external editor)
- Reassess the project

2.4. Overview of the Standalone User Interface Tool

The standalone interface developed in the context of ACCESSIBLE acts as an integrated assessment environment for all the assessment processes supported by ACCESSIBLE. More specifically, Figure 5 presents that start-up screen of the tool that offers a number of different options:

- File Menu, Toolbar and Quick start widget
 - Create new ACCESSIBLE assessment project
 - Open existing project
- Recent Projects widget: Open one of the recently created or opened projects
- Getting started widget: Get help about getting started with the tool

The typical steps involved in performing an evaluation through the standalone tool are

- Create an assessment project
- Add applications to be assessed
- Preview applications and their source
- Edit Project evaluation criteria
- Perform the assessment

- Access assessment results and fix the errors inline (using the embedded code editors)
- Reassess the project

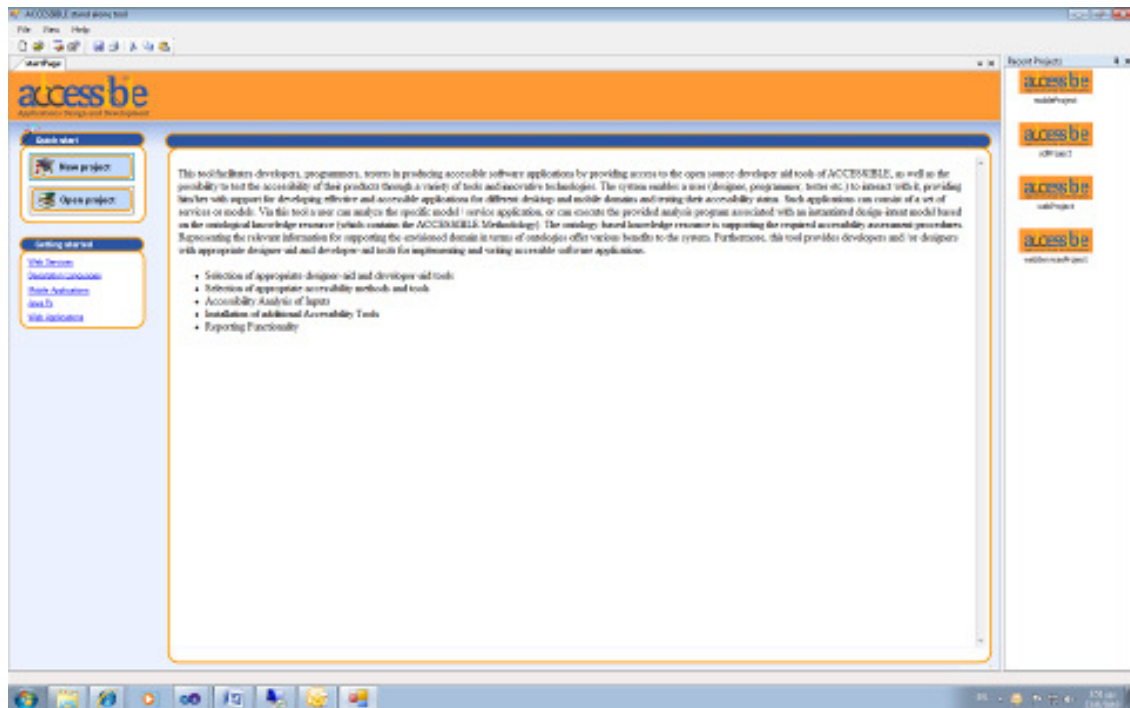


Figure 5: Start-up screen

A detailed description of the standalone tool and its user manual is presented in deliverable D5.1 “User Interface Portal & User Assistant Agent” section 7.2. In the following the four different assessment modules will be described.

3. Web Accessibility Assessment Tool

The Web Accessibility Assessment Tool provides users (developers, designers, testers) with the opportunity to evaluate the accessibility status of a web application according to the WCAG 2.0 and WAI-ARIA guidelines. WaaT can detect accessibility violations and it can also provide informative tips that will aid the users in the correction of the detected errors/warnings.

The implementation of WaaT was initially based on the ACCESSIBLE Harmonised Methodology (HAM) as described in deliverable D3.1, that enabled a personalized accessibility assessment process, though the selection of different accessibility constraints (e.g. different types of impairments and disabilities, different sets of guidelines, personas).

The WaaT consists of the following six main components which are described with more details in the deliverable D5.2 and are being presented in the Figure 6 below:

1. A Graphical User Interface (GUI): The GUI of the WaaT tool consists of a set of forms that can be accessed by users, in order to help them with the accessibility assessment of preferable web applications. The GUI of the assessment tool is also responsible for the representation of the assessment results after the completion of the evaluation process. Via the GUI, the user is able to select the preferable tests to be executed using knowledge concerning the HAM methodology, which is included the ACCESSIBLE ontologies. Moreover, using the GUI, the user can load a Virtual User Model, on which the evaluation process will be based. After the completion of the evaluation process, the Web Accessibility Evaluator returns the results to the GUI, so as to be presented to the user.

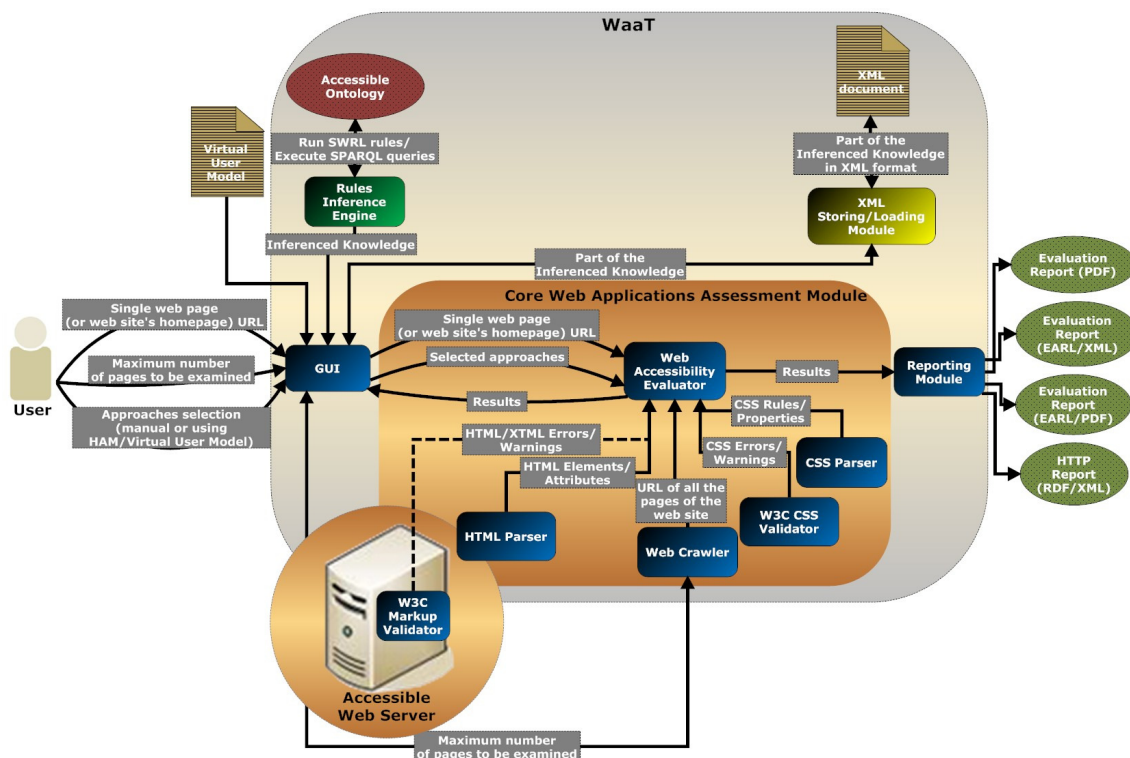


Figure 6: WaaT architecture overview

2. The Rules Inference Engine: The Rules Inference Engine is responsible for the communication between the WaaT and the ACCESSIBLE ontologies where knowledge regarding the HAM methodology is stored. The Rules Inference Engine is able to run SWRL rules as well as execute SPARQL queries, in order to extract specific knowledge from the ontology.

3. The XML Storing/Loading Module: As the execution of SWRL rules defined in the ACCESSIBLE ontology is generally a time-consuming process, an XML storing/loading module is introduced. This module is able to automatically generate an XML file containing all the necessary knowledge of the ontology that is required for the assessment process. After the generation of the XML document the XML Storing/Loading module is responsible for the “virtual connection” between the WaaT tool and the ontology.

4. The Core Web Applications Assessment Module: The Core Web Applications Assessment Module is the core component of the WaaT tool and includes all the required algorithms and methodologies for the execution of preferable user’s selections. This module consists of the following six sub-components:

- i. The **Web Crawler**, which returns the single pages of a web site, when an entire web site is being selected to be evaluated.
- ii. The **W3C Markup Validator**¹, which evaluates the conformance of a web page against the technical specifications of HTML and XHTML.
- iii. The **HTML Parser** that parses the web page source code and returns the necessary information concerning the desired elements/attributes of the HTML/XHTML.
- iv. The **W3C CSS Validator**, which evaluates the conformance of a CSS file against the CSS 2.1 Specifications².
- v. The **CSS Parser** that is responsible for the parsing of all the CSS files that are connected with the examined web page.
- vi. The **Web Accessibility Evaluator**, which is the core component of the Core Web Applications Assessment Module performing the accessibility evaluation of a web application, according to the user’s preferences.

5. The Reporting Module

The Reporting Module of WaaT is responsible for the generation of various types of reports including the results of the evaluation process. The supported report types include the following:

- A PDF report containing all the errors and warnings of the evaluated web page(s). For each error/warning, a description, a tip for its fixation and the corresponding HTML source code (where the violation appears) are provided. Moreover, the success criterion and the technique of WCAG 2.0 (where the error/warning refers to) is also provided.
- An EARL-based report of the detected errors and warnings. EARL is a standard format for support of the evaluation of Web applications. It contains a vocabulary to describe the results of a test’s execution, in order to facilitate its processing and interpretation by different tools. EARL is expressed in

¹ <http://validator.w3.org/>

² <http://www.w3.org/TR/CSS2/>

RDF, which can be extended easily and be adapted to any domain, as in this case accessibility.

- A PDF version of the EARL-based report, containing all the information of the EARL-based report in human readable format.
- A report containing all the HTTP content transferred between the WaaT and the server that hosts the evaluated web application, using the HTTP Vocabulary in RDF. This report can be used as input to other web accessibility evaluators, in order to perform a similar use case scenario and compare their results with those of WaaT.

3.1. Web Accessibility Assessment Tool Implemented Tests

The Web Accessibility Assessment Tool (Waat) supports the accessibility assessment of Web applications according to WCAG 2.0 and WAI-ARIA implemented tests. The following Table 1 presents all the trivial tests that have been implemented in the WaaT, according to the WCAG 2.0 guidelines. More details about the implemented tests and the supported steps are being provided in the following Table 2.

WCAG2 Guideline	Success Criterion	Technique(s)
1.1	1.1.1 (Level A)	H24, H45, H46, H35, H37, H36, H44, H2, H67
	1.2.8 (Level AAA)	H46
1.3	1.3.1 (Level A)	H71, H63, H44, H39, H43, H51, H71, H44, H73, H65
	1.4.2 (Level A)	G170, G18
	1.4.4 (Level AA)	C12, C13, C14, C28
	1.4.6 (Level AAA)	G17, G18
	1.4.8 (Level AAA)	C21, C24, C12, C13, C14
	1.4.10 (Level AAA)	G170, G18
2.1	2.1.1 (Level A)	H91, G90, SCR2, SCR20, G202, H71
	2.2.2 (Level A)	G11
	2.2.4 (Level AAA)	G75, G76
2.3	2.3.1 (Level A)	G15, G19, G176
	2.3.2 (Level AAA)	G19
2.4	2.4.1 (Level A)	H64, H50
	2.4.2 (Level A)	G88, H25, H24
	2.4.5 (Level AA)	G126, H59, G185, G126
	2.4.6 (Level AA)	G130
	2.4.8 (Level AAA)	H59
	2.4.9 (Level AAA)	H30, H24
3.1	3.1.1 (Level A)	H57, H54
	3.1.4 (Level AAA)	H28
3.2	3.2.2 (Level A)	G80
	3.2.5 (Level AAA)	G76, H76, SVR1, H83, G110, SCR24
	3.3.2 (Level A)	G162, H65, H44, H71, H44, G162
	3.3.5 (Level AAA)	H89
4.1	4.1.1 (Level A)	H93, G134, G192, H94, G134, G192, H74, H75, H88
	4.1.2 (Level A)	H91, H65, H64, H88, H44

Table 1: Implemented tests in WaaT following accurately the test procedure defined by the guidelines of WCAG 2.0

WCAG2 Guideline	Success Criterion	Technique	Step(s)	Test description	Implementation details
1.1	1.1.1 (Level A)	G73	1	Check for the existence of an <A> element immediately before or after the elements, when the elements do not have a «longdesc» attribute	G73 refers to non-text content in general. The specific test examines images, as it is non-text content.
		H30	1	Check if images used as hyperlinks (included in an <A> element) have text alternative or text (contained in the <A> element) describing the image	H30 refers to non-text content in general. The specific test examines images, as it is non-text content.
		H86	2	Check for the existence of emoticons	All the emoticons described here: http://www.techdictionary.com/emoticon.html are examined.
		G95	1	Check for the existence of text equivalent in <OBJECT> elements	G95 refers to non-text content in general. The specific test examines <OBJECT> elements.
		G95	1	Check for the presence of non empty “alt” attribute for and <APPLET> elements and for <APPLET> and <OBJECT> elements with text equivalents	G95 refers to non-text content in general. The specific test examines , <APPLET> and <OBJECT> elements.
		H86	2	Check for the existence of ASCII art (enclosed in <PRE> and <XMP> element)	The implementation of this test was based on the algorithm presented here: http://www.w3.org/WAI/ER/IG/ert/AsciiArt.htm
	1.2.3 (Level A) & 1.2.5 (Level AA) & 1.2.7 (Level AAA)	SM11	3	Check whether video freezes in places and plays extended audio description	Check for the existence of videos without extended audio description (“dur” attribute value <= (“clip-end” attribute value – “clip-begin” attribute value) as well as the value of “fill” attribute is “freeze”)
1.3	1.3.2 (Level A)	H34	1	Check for text direction changes	<BDO> elements with “dir” attribute whose value is rtl as well as <BDO> elements with “dir” attribute whose value is ltr are identified.

WCAG2 Guideline	Success Criterion	Technique	Step(s)	Test description	Implementation details
		H34	2	When text changes direction (from left-to-right to right-to-left), check whether punctuation characters occur adjacent to text that is rendered in the non-default direction	Number of <BDO> elements with “dir” attribute whose value is “rtl” that have punctuation characters at the right are identified.
	1.4.3 (Level AA)	G148	4	Check that no background color or image used as a background is specified	The implemented test examines the following: “background” attribute of <body> elements, “bgcolor” attribute of <body> elements, “bgcolor” attribute of <table> elements, “bgcolor” attribute of <tr> elements, “bgcolor” attribute of <td> elements, “bgcolor” attribute of <th> elements, “background” attribute of <style> elements, “background-color” attribute of <style> elements, “background-image” attribute of <style> elements, “background” css properties, “background-color” css properties, “background-image” css properties
	1.4.8 (Level AAA)	C19	1	Check the value of «text-align» .css property	“text-align” properties in CSS with value “center” or “inherit” or “justify” are considered as errors
2.1	2.1.1 (Level A)	G90, SCR20, G202	G90 1, SCR20 2, G202 2	Check for the existence of redundant input mechanisms	The following are identified and considered as errors: • onkeyup event handlers without redundant onmouseup event handlers • onkeydown event handlers without redundant onmousedown event handlers • onmousedown event handlers without redundant onkeydown event handlers • onclick event handlers without redundant onkeypress event handlers • onkeypress event handlers without redundant

WCAG2 Guideline	Success Criterion	Technique	Step(s)	Test description	Implementation details
					onclick event handlers • onmouseup event handlers without redundant onkeyup event handlers
2.4	2.4.3 (Level A)	H4	1	Check for tabbing order among elements (existence of «tabindex» attribute)	The following are examined: • <OBJECT> elements without «tabindex» attribute • <AREA> elements without «tabindex» attribute • <SELECT> elements without «tabindex» attribute • <A> elements without «tabindex» attribute • <TEXTAREA> elements without «tabindex» attribute • <BUTTON> elements without «tabindex» attribute • <INPUT> elements without «tabindex» attribute
	2.4.4 (Level A)	H30	1	Check if images used as hyperlinks (included in an <A> element) have text alternative or text (contained in the <A> element) describing the image	H30 refers to non-text content in general. The specific test examines images, as it is non-text content.
3.1	3.1.3 (Level AAA)	H40	1 & 2 & 3	Check for the existence of definition lists and examine their structure	<DT> elements without a <DD> element immediately following are examined

Table 2: Implemented tests in WaaT including also some assumptions on the test procedure defined by the WCAG 2.0 guidelines

Similarly, the following Table 3 presents all the implemented tests concerning the WAI-ARIA specifications.

WAI-ARIA Test	Implementation details
Check for role attribute presence	Identify elements with “role” attribute and value equal to one of the following: “status”, “tree”, “row”, “alertdialog”, “navigation”, “option”, “menuitem”, “application”, “definition”, “list”, “tabpanel”, “treeitem”, “contentinfo”, “combobox”, “separator”, “note”, “region”, “tablist”, “tooltip”, “log”, “search”, “link”, “checkbox”, “math”, “dialog”, “heading”, “document”, “radiogroup”, “rowheader”, “banner”, “img”, “group”, “progressbar”, “main”, “marquee”, “menubar”, “listbox”, “presentation”, “radio”, “treegrid”, “directory”, “complementary”, “button”, “menu”, “rowgroup”, “alert”, “toolbar”, “grid”, “menuitemcheckbox”, “menuitemradio”, “spinbutton”, “tab”, “textbox”, “scrollbar”, “timer”, “slider”, “gridcell”, “form”, “columnheader”, “listitem”, “article”
Check for elements with missing required WAI-ARIA properties	Examine the following (based on http://www.w3.org/TR/wai-aria/roles): • Elements with role = spinbutton without attribute aria-valuemax • Elements with role = treeitem whose parent is not group or tree • Elements with role = tab whose parent is not tablist • Elements with role = application including more than one elements with role = banner. • Elements with role = row whose parent has not role grid or row group or tree grid • Elements with role = document including more than one elements with role = banner. • Elements with role = tree whose children are not groups or tree items • Elements with role = application missing title and aria-labelledby attributes. • Elements with role = option whose parents are not elements with role = combobox or role = listbox or role = menu or role = radiogroup or role = tree • Elements with role = progress bar missing aria-valuemax attribute • Elements with role = tablist whose children are not tabs • Elements with role = radiogroup whose parents are not elements with role = radio • Elements with role = combobox not including elements with role = listbox or role = textbox. • Elements with role = list whose children are not elements with role = listitem or role = group • Elements with role = spinbutton without attribute aria-valuenow • Elements with role = navigation missing the aria-labelledby attribute • Elements with role = dialog missing aria-label and aria-labelledby attributes. • Elements with role = alertdialog whose aria-describedby attribute is missing

WAI-ARIA Test	Implementation details
	• Images with role = presentation having also non empty alt attribute • Elements with role = tabpanel without aria-labelledby attribute and without aria-controls property on the corresponding tab • Elements with role = menuitem whose parent has not role = menu or role = menubar • Elements with role = checkbox without attribute aria-checked • Elements with role = rowgroup whose children are not rows • Elements with role = scrollbar without attribute aria-valuenow • Elements with role = rowheader whose parent has not role = row • Role document incorrectly applied on element other than body. • Elements with role = scrollbar without attribute aria-valuemin • Elements with role = rowgroup whose parent has not role = grid • Elements with role = column header not contained or owned by an element with role = row. • Elements with role = content info without aria-labelledby attribute • Elements with role = gridcell not owned by elements with role = row • Elements with role = complementary without aria-labelledby attribute • Elements with role = combobox without attribute aria-autocomplete. • Elements with role = application containing more than one elements with role = main • Elements with role = slider without attribute aria-valuenow • Elements with role = menuitemradio whose parent has not role = menu or role = menubar • Elements with role = menuitemcheckbox whose parent has not role = menu or role = menubar • Elements with role = spinbutton without attribute aria-valuemin • Elements with role = scrollbar without attribute aria-valuemax • Elements with role = slider without attribute aria-valuemax attribute • Elements with role = menu whose children are not elements with role = group or role = menuitem or role = menuitemcheckbox or role = menuitemradio • Elements with role = slider without attribute aria-valuemin • Elements with role = progress bar missing aria-valuemin attribute • Elements with role = listbox whose children are not elements with role = option • Elements with role = radio without aria-checked attribute

WAI-ARIA Test	Implementation details
	• Elements with role = scrollbar without attribute aria-orientation • Elements with role = toolbar missing aria-labelledby property while there are more than one toolbars • Elements with role = menuitemradio without aria-checked attribute • Elements with role = form missing aria-labelledby attribute • Elements with role = row whose children are not column header or gridcell or row header • Elements with role = menuitemcheckbox without attribute aria-checked • Elements with role = region missing aria-labelledby attribute or having aria-labelledby attribute but not referencing a heading • Elements with role = document including more than one element with role = contentinfo. • Elements with role = combobox without attribute aria-expanded • Elements with role = search without the aria-labelledby attribute • Elements with role = document containing more than one elements with role = main • Elements with role = treegrid whose children are not rows • Elements with role = application including more than one elements with role = content info. • Elements with role = scrollbar without attribute aria-controls • Elements with role = document missing title and aria-labelledby attributes • Role application incorrectly applied on element other than the body.
Check for problems in aria properties	Examine the following: • Elements with id ref specified in more than one elements' aria-owns attribute • Elements with aria-labelledby property referencing an ID that does not exist • Elements with aria-describedby property referencing an ID that does not exist • Elements with aria-owns property referencing an ID that does not exist
Check for elements with invalid role attribute	Elements with invalid role, which is not one of the following: "alert", "alertdialog", "application", "article", "banner", "button", "checkbox", "columnheader", "combobox", "complementary", "contentinfo", "definition", "dialog", "directory", "document", "form", "grid", "gridcell", "group", "heading", "img", "link", "list", "listbox", "listitem", "log", "main", "marquee", "math", "menu", "menubar", "menuitem", "menuitemcheckbox", "menuitemradio", "navigation", "note", "option", "presentation", "progressbar", "radio", "radiogroup", "region", "row", "rowgroup", "rowheader", "search", "separator", "scrollbar", "slider", "spinbutton", "status", "tab", "tablist", "tabpanel", "textbox", "timer", "toolbar", "tooltip", "tree", "treegrid", "treeitem"

Table 3: Tests implemented in the WaaT according to the WAI-ARIA specifications

3.2 Description of Web Accessibility Assessment Tool functionalities (through the ACCESSIBLE Portal)

The main page of the ACCESSIBLE portal is depicted in Figure 7. The Web Accessibility Assessment Tool can be accessed by clicking on the “Website Accessibility Inspection Projects”.



Figure 7: Web Accessibility Inspection Projects

In the next step, the active projects are available. The user can either assess an already added project, or he/she can add a new project (Figure 8).

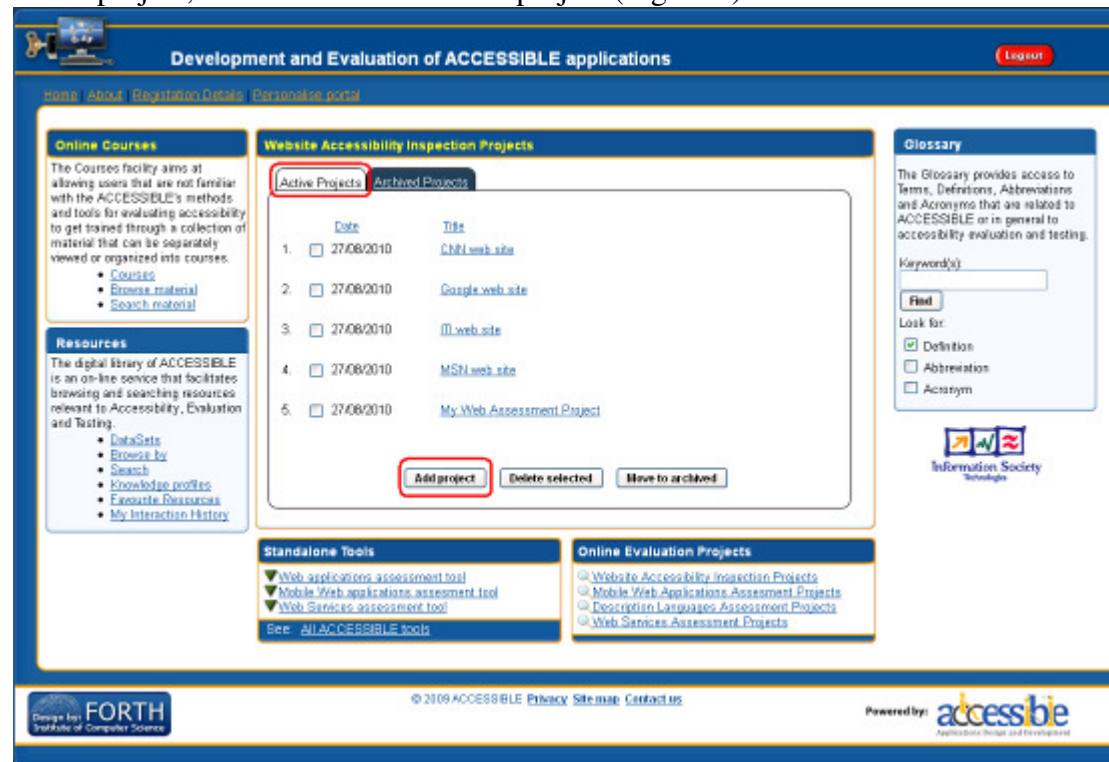


Figure 8: Add a new project

If the user decides to add a new project, then he/she has to specify the title, the description and the URL of the new project (Figure 9). Finally, he/she can continue with the assessment of the new project.

The other offered functionality is to assess a project that is already added. The user has to select the project he/she wishes to assess and then Figure 10 appears. The user has five different options to assess the preferred web site. These options are the same as in the standalone version of the Web Accessibility Assessment Tool and are the following:

- *Evaluate the Web Site against Personas*
- *Evaluate the Web Site against Ontology*
- *Quick evaluation against Ontology*
- *Evaluate the Web Site against specific guidelines*
- *Quick evaluation against specific guidelines*

Add new Web Assessment Project

[Back](#)

Title:

Paragraph Font Size Color Highlight

B *I* U abc x² x₂ [List Icons] [Table Icon] [Link Icon] [Image Icon]

[Undo] [Redo] [Find] [Symbols]

Enter project description

Design HTML

Selected Files(up to 20 MB)

2. File title:

url:

Figure 9: Details concerning a new project

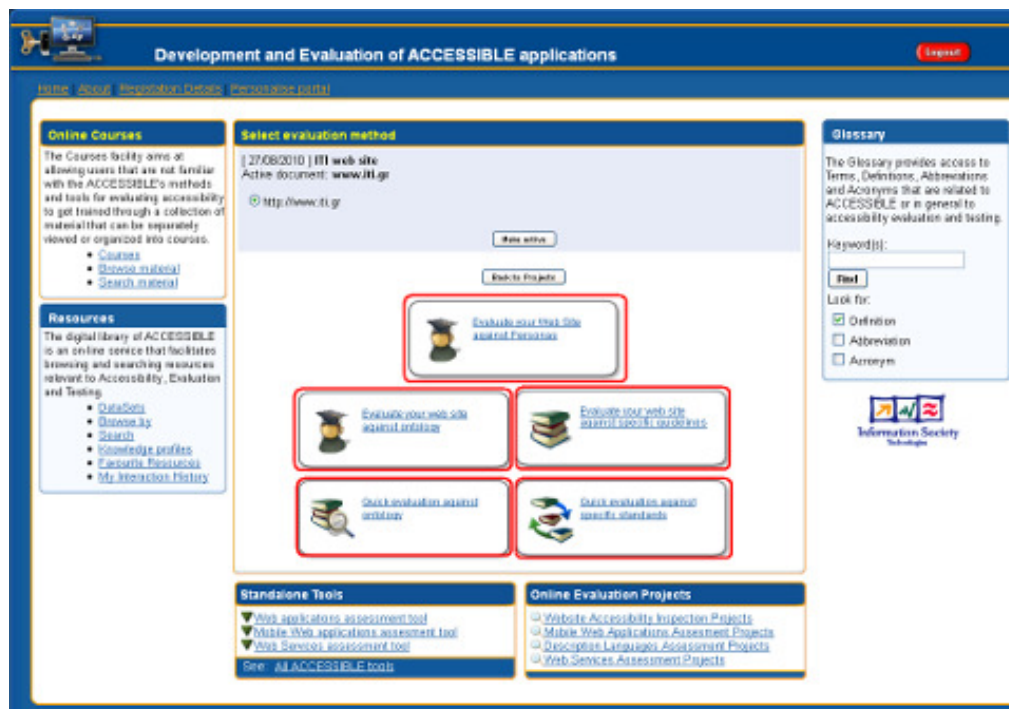


Figure 10: Five options for the assessment of a Web application

Evaluating a Web Site against Personas

By clicking in the first option, the user chooses to evaluate the Web Site against Personas. The first step of the procedure is depicted in Figure 11. All the needed information about the Personas is presented to the user. Then, the user can choose against which Persona, he/she will perform the evaluation process. All the information about the Personas is also available in PDF format.



Figure 11: Evaluation against Personas, step 1

Step 2 includes the review of the corresponding evaluation approaches (Figure 12). The user can select which approaches will be activated during the evaluation process. By pressing the “*Next*” button, he/she proceeds to step 3.

Step 2 of 3: Review Evaluation Approaches

[27/08/2010] ITI web site

Document for assessment: <http://www.iti.gr>

The following evaluation approaches have been inferred for the selected settings:

WCAG2 Evaluation Approaches

- ☒ Check if the "alt" attribute for elements is empty
- ☒ Check if there is no "alt" attribute for elements
- ☒ Check for the existence of "title" attribute for <FRAME> elements
- ☒ Check for the existence of text equivalent in <OBJECT> elements
- ☒ Check for the existence of "alt" attribute in <APPLET> elements
- ☒ Check for the existence of text equivalent in <APPLET> elements
- ☒ Check for the existence of <TD> elements without "scope" attribute
- ☒ Check for the existence of <TH> elements without "scope" attribute
- ☐ Check for the existence of <H1> elements
- ☐ Check for the existence of <H2> elements
- ☐ Check for the existence of <H3> elements
- ☐ Check for the existence of <H4> elements
- ☐ Check for the existence of <H5> elements
- ☐ Check for the existence of <H6> elements
- ☒ Check for the existence of redundant text links for each active region of a server-side image map within the body of an <OBJECT> element
- ☒ Check for the existence of alternate text of elements with "ismap" attribute
- ☒ Check for the existence of "refresh" <META> elements
- ☒ Check for the existence of <A> elements with blank target (target="_blank") and not containing the phrase "opens new window"

Figure 12: Evaluation against Personas, step 2

Figure 13 presents the final step of the evaluation process which includes the Evaluation Report, containing all the errors and the warnings that were detected for the specific Web Site.

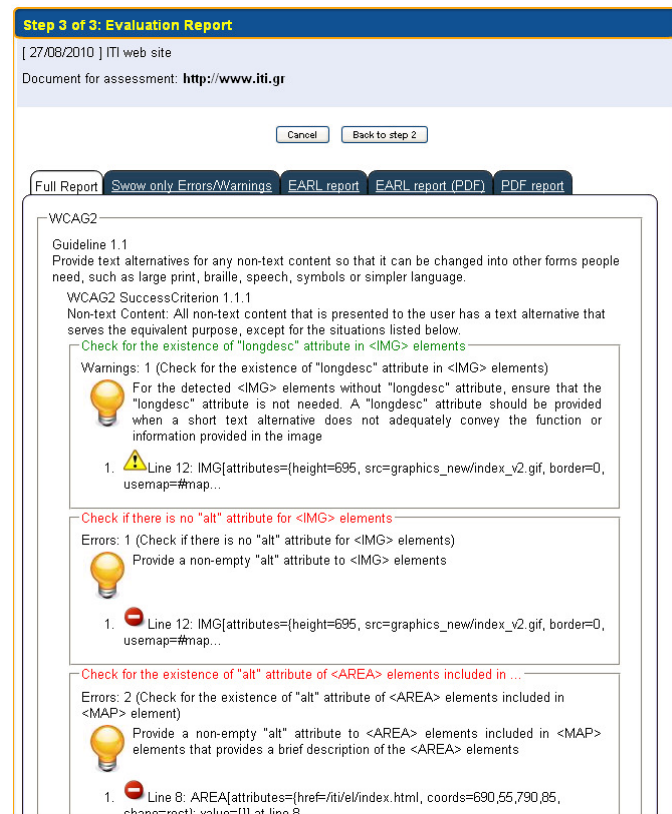


Figure 13: Evaluation against Personas, step 3

The results can be presented to the user with five different ways. These are:

- *Full Report*
- *Show only Errors/Warnings*
- *EARL report*
- *EARL report (PDF)*
- *PDF report*

Figure 13 depicts the *Full Report* and Figure 14 depicts *only the detected Errors/Warnings*. In Figure 15 the results are presented to the user using the EARL format.

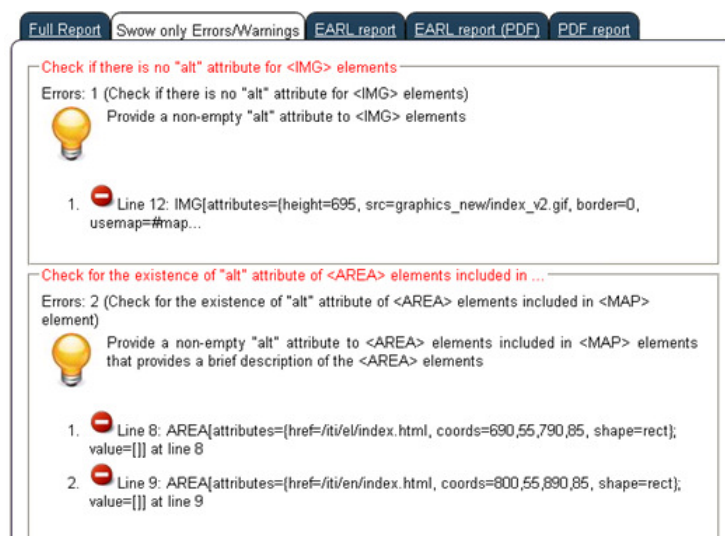


Figure 14: Report containing only errors and warnings

The Evaluation and Report Language (EARL) 1.0 [1] is a vocabulary, developed mainly for the facilitation of the exchange of test results between Web accessibility evaluation tools. For this reason, the EARL report is generated in an XML format, as depicted in Figure 15. Due to the fact, that not all users are familiar with the XML format, the same EARL report is also available in a human-readable PDF format, as depicted in Figure 16. More information about the implementation of the EARL-based Reporting tool, the related technologies and the architecture are available in the Deliverable D5.4 “EARL-based reporting Tool”.



Figure 15: Report using the EARL format

Full Report Summary Errors/Warnings EARL report EARL report (HTML) PDF report	
Human readable diagnosis result based on the EARL report created according to the	
Evaluation and Report Language (EARL) 1.0 Schema W3C Working Draft 26 October 2009	
TestSubject	http://www.iti.gr
Title	Informatics and Telematics Institute
Date	Mon, 06 Jun 2011 13:15:51
Score	47/65
Software	http://www.it-gr/accessible/WCAG2.0_WebAssessmentTool.zip
Title	WCAG2.0 Web Assessment Tool
Description	WCAG2.0 Web Assessment Tool is developed by CERTH/ITI within the premises of the ACCESSIBLE project
Success Criterion	1.1.1
Level	A
Mode	automatic
Outcome	failed
Technique	G95
Technique Step	1
Technique Description	Providing short text alternatives that provide a brief description of the non-text content
Technique Tests	G95 test procedure
Error Description	Number of elements with null "alt" attribute
Problematic Element	<code>; value=[]</code> at line 12
Tip	Provide a non-empty "alt" attribute to elements

Figure 16: Report in a human-readable EARL format (PDF document)

Finally, a PDF report is generated, containing all the necessary information and is also available to the user. The PDF report is depicted in Figure 17.

Full Report Summary Errors/Warnings EARL report EARL report (HTML) PDF report				
Analysis of the detected errors				
Success Criterion	Technique	Error description	Tip	Problematic element
Guideline 1.1 - Level A				
1.1.1 (automatic)	G95 (Step 1) & H37 (Step 2)	Number of elements with null "alt" attribute	Provide a non-empty "alt" attribute to elements	<code>; value=[]</code> at line 12
1.1.1 (automatic)	G73 (Step 1)	Number of elements without adjacent <A> element	Provide an <A> element immediately before or after the elements. Ensure that the <A> element is a valid link that points directly to the long description of the elements. Ensure that the long description conveys the same information as the elements and check for the availability of a link or back function to get the user back to the original location of the elements	<code></code> at line 12
1.1.1 (automatic)	G95 (Step 1) & H24 (Step 1)	Number of total <ARLA> elements included in <MAP> elements without "alt" attribute	Provide a non-empty "alt" attribute to <ARLA> elements included in <MAP> elements that provides a brief description of the <ARLA> elements	<code>ARLA[alt="" href=/index.html, coords=690,65,790,95, shape=rect]; value=[]</code> at line 8

Figure 17: Report in PDF format

Evaluating a Web Site against Ontology

The second option is to evaluate a Web Site against Ontology. More specifically to evaluate the Web Site using the Harmonized methodology, which is described in D3.1 “ACCESSIBLE harmonizes methodology”. By clicking on the button “*Evaluate you Web site against Ontology*”, the Disabilities are presented to the user. Then, the user has the opportunity to choose between the Disabilities that are included in the Harmonized methodology. By these means, the user can conduct a personalized evaluation. Moreover, the user can choose either an Impairment or an Assistive Device that is used by a specific group of people. This procedure is depicted in Figure 18.

Step 1 of 3: Select Evaluation criteria

[27/08/2010] ITI web site
Document for assessment: <http://www.iti.gr>

Cancel Back to project options Next

Disability Functional Limitation Impairment ATs Software

- ☐ Night blindness
- ☐ Deafness
- ☐ Cerebral palsy
- ☐ DeafBlindness
- ☐ Arthritis
- ☐ Asperger's syndrome
- ☐ Quadriplegia
- ☐ Dyspraxia
- ☐ Speech and language disorders
- ☐ Multiple sclerosis
- ☐ Academic skills disorders
- ☒ Blindness
- ☒ Extreme light sensitivity
- ☒ Attention Deficit Hyperactivity Disorder
- ☐ Mixed Hearing Loss
- ☐ Dysarthria
- ☐ Down syndrome
- ☐ Loss of central vision
- ☐ Communication disability
- ☐ Absent limb/ reduced limb function

Figure 18: Evaluation against Ontology, step 1

By pressing the button “*Next*”, the corresponding approaches are presented to the user (Figure 19). These approaches derive from the Harmonized methodology and are the approaches that affect the Disabilities that were selected in Step 1.

Step 2 of 3: Review Evaluation Approaches

[27/08/2010] ITI web site
Document for assessment: <http://www.iti.gr>

[Cancel](#) [Back to step 1](#) [Next](#)

The following evaluation approaches have been inferred for the selected settings:

WCAG2 Evaluation Approaches

- ☒ Check if the "alt" attribute for elements is empty
- ☒ Check if there is no "alt" attribute for elements
- ☒ Check for the existence of "title" attribute for <FRAME> elements
- ☒ Check for the existence of text equivalent in <OBJECT> elements
- ☒ Check for the existence of "alt" attribute in <APPLET> elements
- ☒ Check for the existence of text equivalent in <APPLET> elements
- ☒ Check for the existence of <TD> elements without "scope" attribute
- ☒ Check for the existence of <TH> elements without "scope" attribute
- ☐ Check for the existence of <H1> elements
- ☐ Check for the existence of <H2> elements
- ☐ Check for the existence of <H3> elements
- ☐ Check for the existence of <H4> elements
- ☐ Check for the existence of <H5> elements
- ☐ Check for the existence of <H6> elements
- ☒ Check for the existence of redundant text links for each active region of a server-side image map within the body of an <OBJECT> element

Figure 19: Evaluation against Ontology, step 2

Finally, by pressing the button “Next” the results are presented to the user, as it is obvious in Figure 20.

Full Report [Show only Errors/Warnings](#) [EARL report](#) [EARL report \(PDF\)](#) [PDF report](#)

WCAG2

Guideline 1.1
Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

WCAG2 SuccessCriterion 1.1.1
Non-text Content: All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.

Check for the existence of "longdesc" attribute in elements

Warnings: 1 (Check for the existence of "longdesc" attribute in elements)

 For the detected elements without "longdesc" attribute, ensure that the "longdesc" attribute is not needed. A "longdesc" attribute should be provided when a short text alternative does not adequately convey the function or information provided in the image

1.  Line 12: IMG[attributes={height=695, src=graphics_new/index_v2.gif, border=0, usemap=#map...]

Check if there is no "alt" attribute for elements

Errors: 1 (Check if there is no "alt" attribute for elements)

 Provide a non-empty "alt" attribute to elements

1.  Line 12: IMG[attributes={height=695, src=graphics_new/index_v2.gif, border=0, usemap=#map...]

Check for the existence of "alt" attribute of <AREA> elements included in ...

Errors: 2 (Check for the existence of "alt" attribute of <AREA> elements included in <MAP> element)

 Provide a non-empty "alt" attribute to <AREA> elements included in <MAP> elements that provides a brief description of the <AREA> elements

1.  Line 8: AREA[attributes={href=/iti/el/index.html, coords=690,55,790,85, shape=rect}; value=[]] at line 8

2.  Line 9: AREA[attributes={href=/iti/en/index.html, coords=800,55,890,85, shape=rect}; value=[]] at line 9

Figure 20: Evaluation against Ontology, step 3

Quick evaluation against Ontology

The third option is to perform a quick evaluation against the ontology. By clicking on the button “Quick evaluation against Ontology”, the use can select between the

different categories, as depicted in Figure 21. The next two steps are the same as in the *evaluation against Ontology*. More specifically, the user chooses the approaches and then the results are presented.

Step 1 of 3: Select Categories

[27/08/2010] ITI web site

Document for assessment: <http://www.iti.gr>

Cancel Back to project options Next

Evaluation Categories

- ☐ Disability
- ☐ Functional Limitation
- ☒ Impairment
- Standards
 - ☒ CSS rules
 - ☒ WCAG2 Guidelines
- Devices
 - ☐ Alternative Keyboards or Switches
 - ☐ Braille Displays
- Software
 - ☐ Scanning Software
 - ☐ Screen Magnifiers
 - ☐ Screen Readers
 - ☐ Speech Synthesis & Recognition
 - ☐ Text Browsers

Figure 21: Quick evaluation against Ontology, step 1

Evaluating a Web Site against specific guidelines

The fourth option is to evaluate a Web Site against specific guidelines. By clicking on the button “*Evaluate you Web site against specific guidelines*”, the Guidelines and the Success Criteria of WCAG2.0 are presented to the user, as depicted in Figure 22. The user can select against which Success Criteria he/she prefers to be used during the evaluation process. After the selection of the Success Criteria, the user presses the button “*Next*” and then he/she can review the corresponding approaches, as showed in Figure 23.

Guidelines based settings

[27/08/2010] ITI web site
Document for assessment: <http://www.iti.gr>

Cancel Back to project options Next

WCAG2 CSS

Guideline 1.1
Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

☒ WCAG2 SuccessCriterion 1.1.1
Non-text Content: All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.

Guideline 1.2
Provide alternatives for time-based media.

☒ WCAG2 SuccessCriterion 1.2.1
Audio-only and Video-only (Prerecorded)

☒ WCAG2 SuccessCriterion 1.2.2
Captions are provided for all prerecorded audio content in synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

☒ WCAG2 SuccessCriterion 1.2.3
An alternative for time-based media or audio description of the prerecorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

☒ WCAG2 SuccessCriterion 1.2.4
Captions are provided for all live audio content in synchronized media.

☒ WCAG2 SuccessCriterion 1.2.5
Audio description is provided for all prerecorded video content in synchronized media.

☒ WCAG2 SuccessCriterion 1.2.6
Sign language interpretation is provided for all prerecorded audio content in synchronized media.

☒ WCAG2 SuccessCriterion 1.2.7
Where pauses in foreground audio are insufficient to allow audio descriptions to convey the sense of the video, extended audio description is provided for all prerecorded video content in synchronized media.

Figure 22: Evaluation against specific guidelines, step 1

Step 2 of 3: Review Evaluation Approaches

[27/08/2010] ITI web site
Document for assessment: <http://www.iti.gr>

Cancel Back to step 1 Next

The following evaluation approaches have been inferred for the selected settings:

WCAG2 Evaluation Approaches

- ☒ Check if the "alt" attribute for elements is empty
- ☒ Check if there is no "alt" attribute for elements
- ☒ Check for the existence of "title" attribute for <FRAME> elements
- ☒ Check for the existence of text equivalent in <OBJECT> elements
- ☒ Check for the existence of "alt" attribute in <APPLET> elements
- ☒ Check for the existence of text equivalent in <APPLET> elements
- ☒ Check for the existence of <TD> elements without "scope" attribute
- ☒ Check for the existence of <TH> elements without "scope" attribute
- ☒ Check for the existence of <H1> elements
- ☒ Check for the existence of <H2> elements
- ☒ Check for the existence of <H3> elements
- ☒ Check for the existence of <H4> elements
- ☒ Check for the existence of <H5> elements
- ☒ Check for the existence of <H6> elements
- ☒ Check for the existence of redundant text links for each active region of a server-side image map within the body of an <OBJECT> element
- ☒ Check for the existence of alternate text of elements with "ismap" attribute

Figure 23: Evaluation against specific guidelines, step 2

Finally, by pressing the button “*Next*”, the results of the evaluation process are presented to the user, as depicted in Figure 24. The other four ways of presenting the evaluation results are available for the possible evaluation processes.



Figure 24: Evaluation against specific guidelines, step 3

Quick evaluation against specific guidelines

The third option is to perform a quick evaluation against the ontology. By clicking on the button “*Quick evaluation against specific guidelines*”, the user can select between the available Standards, as depicted in Figure 25. The next two steps are the same as in the *evaluation against specific guidelines*. More specifically, the user chooses the approaches and then the results are presented.



Figure 25: Quick evaluation against specific guidelines, step 1

3.3 Web Accessibility Assessment (using the Standalone Tool)

Web Assessment initiates with the creation of a new ACCESSIBLE web assessment project, as presented in Figure 26, or by opening an existing project by browsing the local file system, as shown in Figure 27.

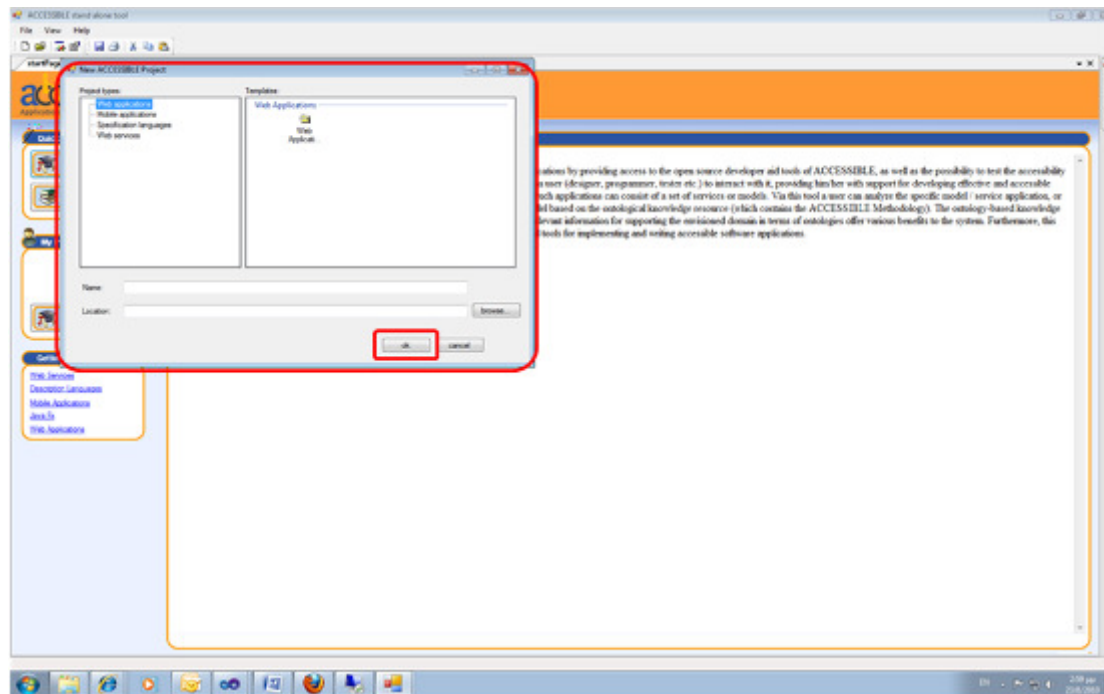


Figure 26: Create new project

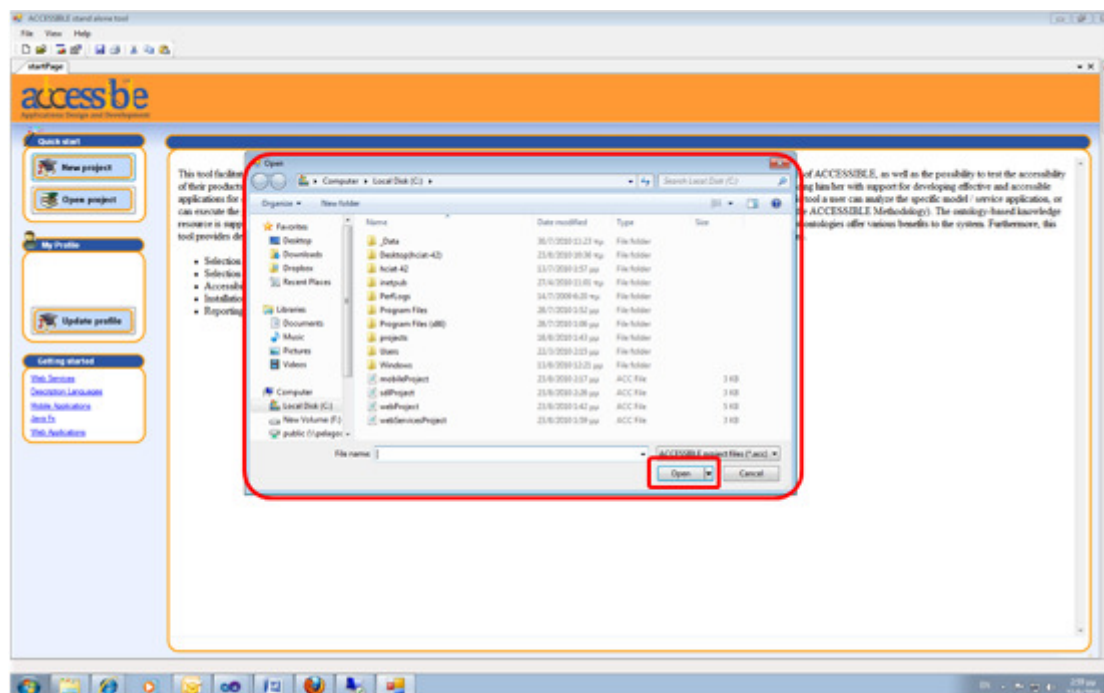


Figure 27: Open ACCESSIBLE project

Files can be added to the project either by selecting a saved web page in the local file system (see Figure 28), or by entering the web page url (see Figure 29) for allowing the integration to the project of pages that reside on the web.

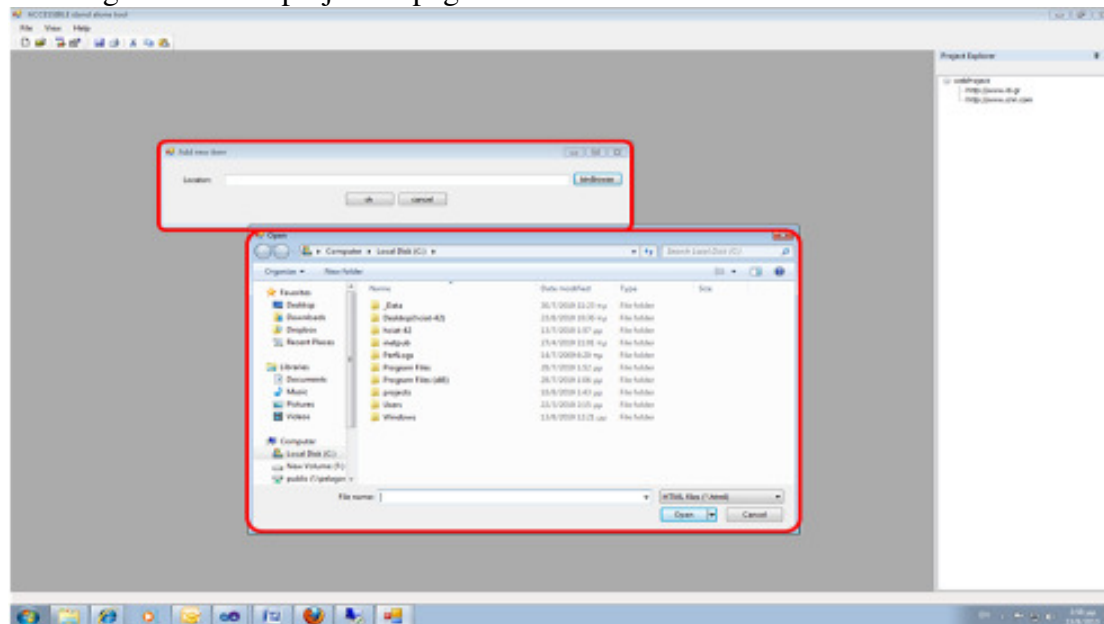


Figure 28: Add new item (file from local file system)

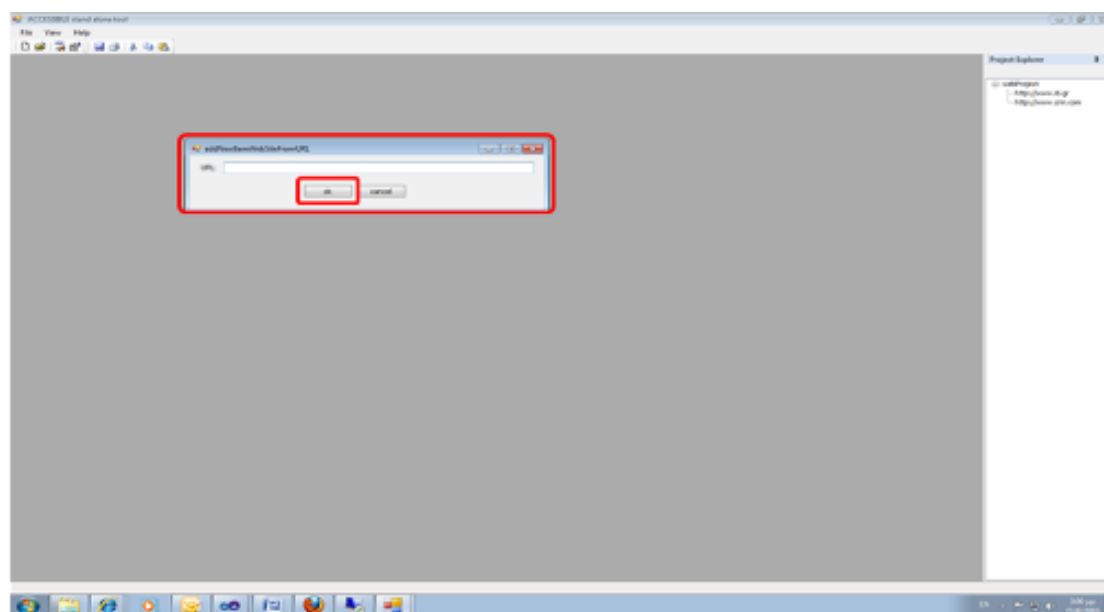


Figure 29: Add new item (web-page from URL)

The added web pages appear on the project explorer widget, offering the option to view the web page's source code (Figure 30) or to browse the web page by using one of the embedded browsers (Internet Explorer or Firefox as presented in Figure 31)

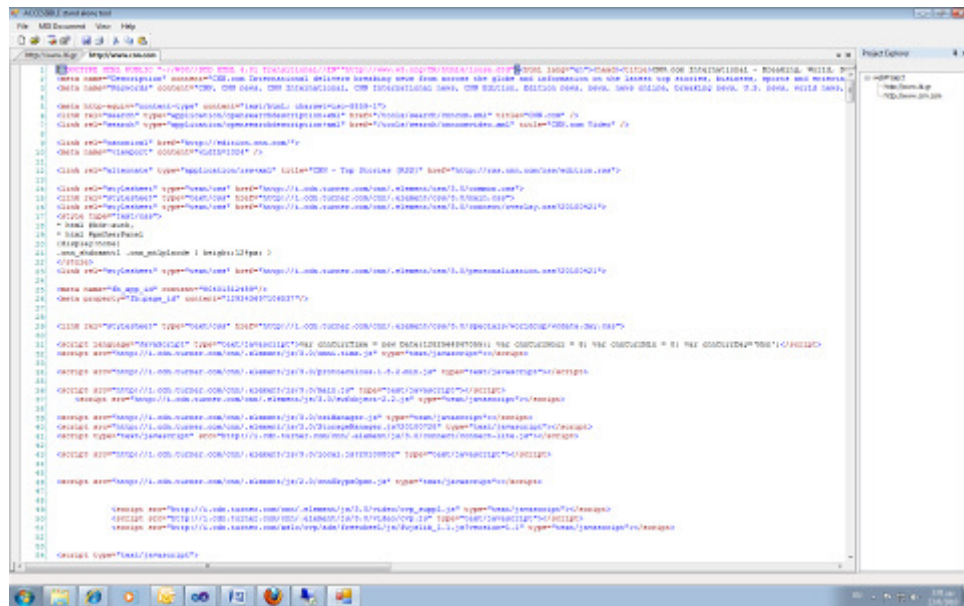


Figure 30: Source view of HTML document

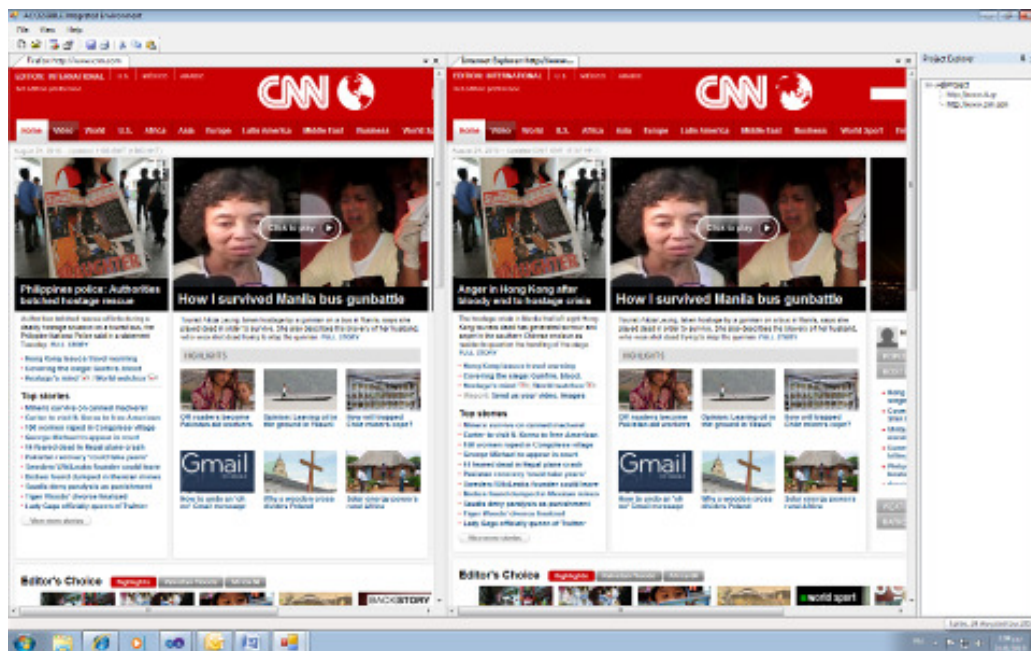


Figure 31: Preview Web Page using one of the embedded browsers (Internet Explorer or Firefox)

Web Assessment Criteria

The selection of the project options menu item from the solution explorer provides access to the assessment criteria of the specific project, as presented in Figure 32. These criteria represent the options offered by the ACCESSIBLE ontology and are used for filtering the evaluation approaches to be used when performing the assessment. For a Web Assessment Project the following categories of assessment criteria are available:

- Ontology
 - Disability
 - Functional Limitation
 - Impairment

- Standard
- Device
- Software
 - Scanning Software
 - Screen Magnifiers
 - Screen Readers
 - Speech Synthesis & Recognition
 - Text Browsers
- Specific Guidelines
 - WCAG2
 - WAI-ARIA
 - CSS
 - Personas

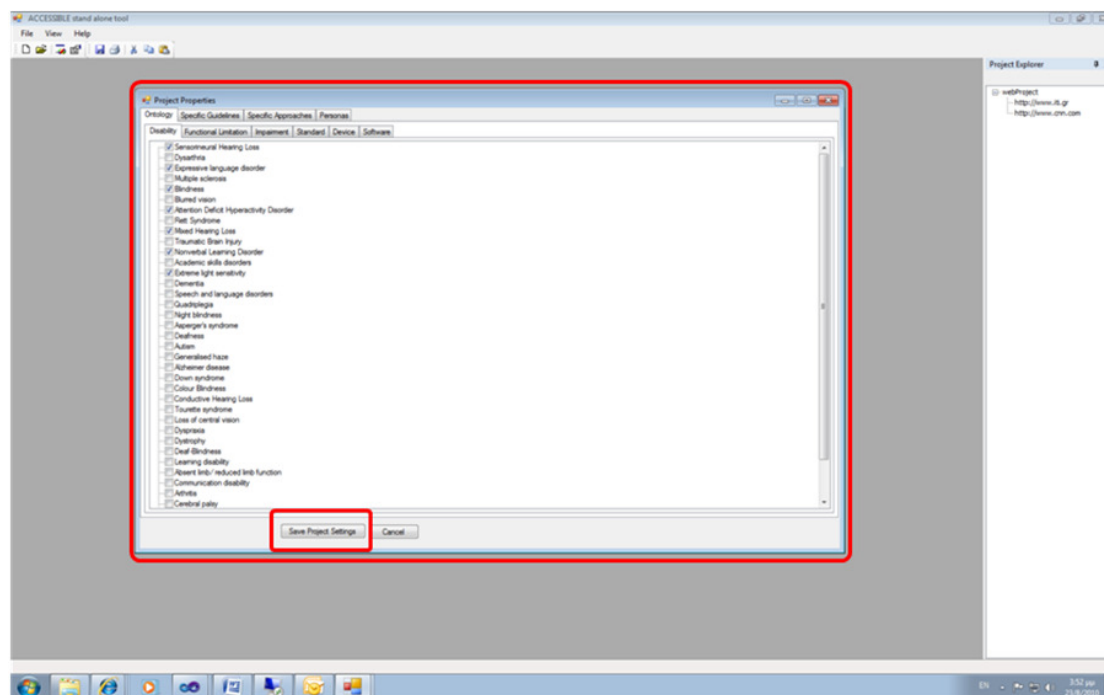


Figure 32: Selection of assessment criteria through the project properties form

Assessment and Reporting

After selection of the web page to be evaluated, the assessment results are presented in a number of different ways:

- For Development purposes
 - Full evaluation output (see Figure 33)
 - Errors List (see Figure 34)
 - Warnings List (see Figure 35)
- For Reporting
 - PDF Report (see Figure 36)
 - EARL Report (see Figure 37)



Figure 33: Web Page Assessment results: Full assessment output



Figure 34: Web Page Assessment results: Errors List



Figure 35: Web Page Assessment results: Warnings List

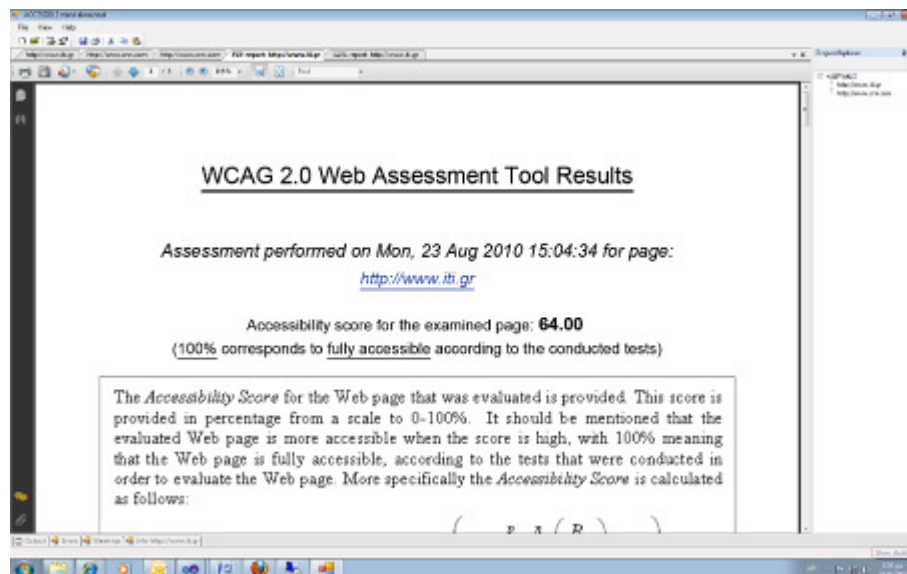
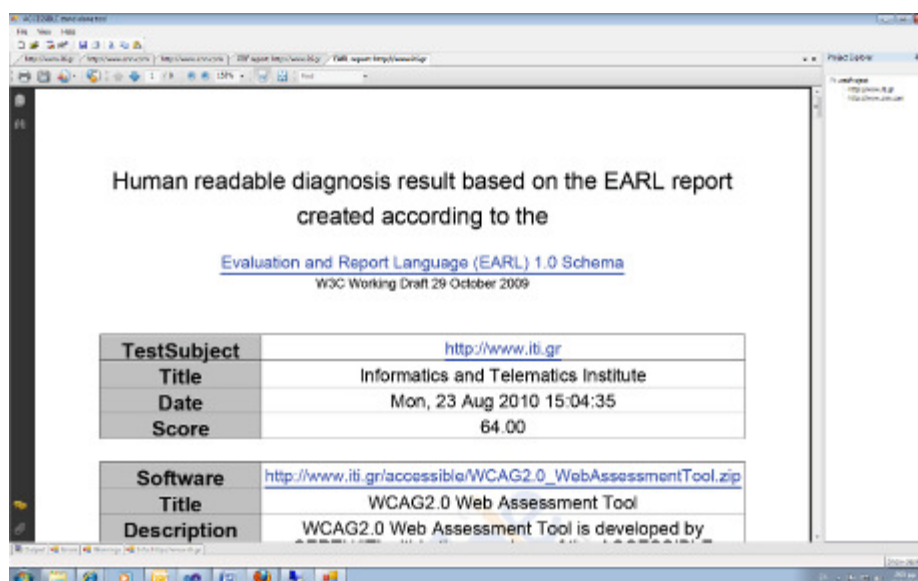


Figure 36: Web Page Assessment results (PDF report)



Human readable diagnosis result based on the EARL report created according to the	
<u>Evaluation and Report Language (EARL) 1.0 Schema</u> W3C Working Draft 29 October 2009	
TestSubject	http://www.iti.gr
Title	Informatics and Telematics Institute
Date	Mon, 23 Aug 2010 15:04:35
Score	64.00
Software	http://www.iti.gr/accessible/WCAG2.0_WebAssessmentTool.zip
Title	WCAG2.0 Web Assessment Tool
Description	WCAG2.0 Web Assessment Tool is developed by

Figure 37: Web Page Assessment results (EARL report in PDF format)

3.4 Changes in current version

Comparing the latest version of Web Accessibility Assessment module with the previous one, a number of changes were made. These changes are based mainly on the comments received during the review of the project as well as the pilot results phase 2 and 3. Thus some of the following important changes were performed:

- A new HTML Parser was integrated, in order to parse the web page source code and get the necessary information concerning the desired elements/attributes of the HTML/XHTML, This parser is the jsoup library[2], which is Java library for working with real-world HTML. It provides a very convenient API for extracting and manipulating data.
- The Accessible Rich Internet Applications (WAI-ARIA) 1.0 was added to the ACCESSIBLE ontology, thus enabling Web Accessibility Assessment tool to provide a basic assessment based on ARIA specification. All the states and properties that are defined within ARIA specification were added to the ontology, thus enabling the tool to have access to them and perform the corresponding tests. Figure 38 shows a part of the instances of WAI-ARIA added to the ACCESSIBLE ontology.

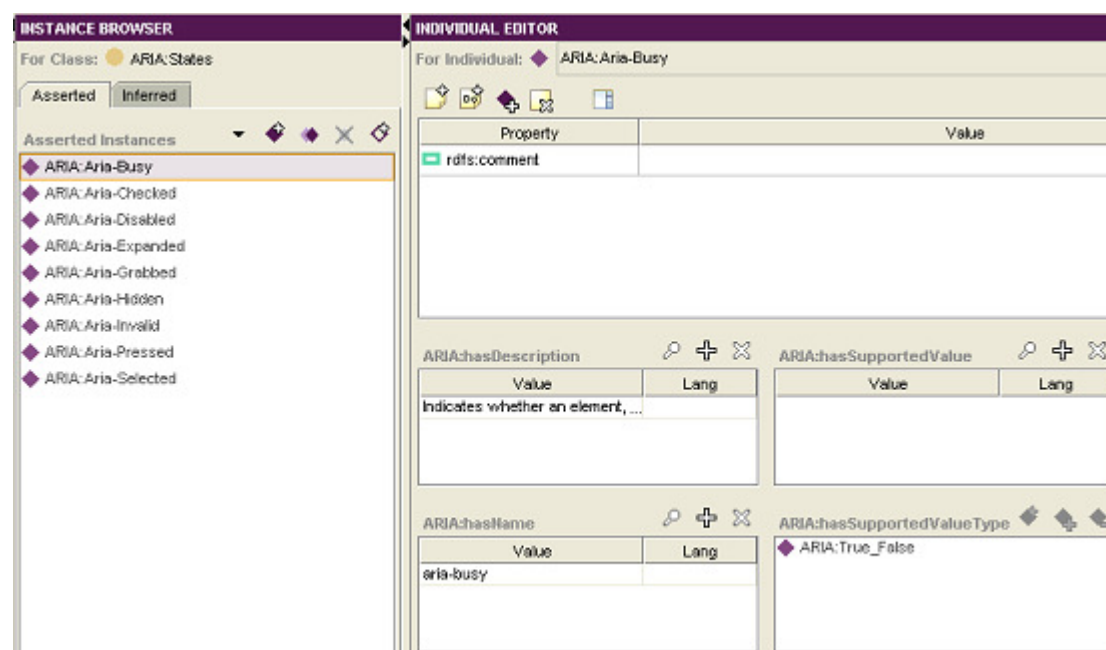


Figure 38: WAI-ARIA ontology instances

- A number of tests have been implemented according to the WAI-ARIA specification, as presented with details in section 3.1.
- The evaluation of upper limb impaired users can be performed through the improved version of the tool.
- The descriptions of the errors as well as the provided hints have been corrected/improved.
- The mechanism that calculates the line in the source code where the error/warning occurs has been improved in the latest version of the tool. Regarding the slow scrolling and other GUI problems, all of them fixed and addressed in the Web version of the tool.
- The generated reports (both EARL and PDF) provide information concerning Personas, when they are used during the evaluation process. Figure 39 depicts the EARL report, in PDF format, generated after an evaluation process concerning the Persona “Emma Karlsson”.

Human readable diagnosis result based on the EARL report
created according to the

[Evaluation and Report Language \(EARL\) 1.0 Schema](#)
W3C Working Draft 28 October 2008

TestSubject	http://www.it-gr
Title	Informatics and Telematics Institute
Date	Mon, 06 Jun 2011 15:15:55
Impairment	Upper Limb Impairment
Impairment	Cognitive Impairment
Impairment	Communication received and producing impairments
Impairment	Vision Impairment
Impairment	Hearing Impairment

Software	http://www.it-gr/accessibility/WCAQ2.0_WebAssessmentTool.zip
Title	WCAQ2.0 Web Assessment Tool
Description	WCAQ2.0 Web Assessment Tool is developed by CERTH/ITI within the premises of the ACCESSIBLE project.
Score	51.29

Persona	Emma Karlsson
hasAge	39
hasMaritalStatus	Single
hasEducation	School of arts
hasJob	Employed
hasLocation	Sollentuna, Sweden
Meet	Emma (39) is Swedish and grew up in Sollentuna, Stockholm. At birth she suffered neurological damage, causing her to have dysarthria. Emma whispers very softly and produces abnormal intonation when speaking, making it hard for other people to understand her. She also has hearing problems when people or devices speak to quickly. Emma works as an independent photographer and regularly gets contracted through her extended network she built throughout the years. Most of the assignments she takes on are weddings or communions. Emma is also an active helper at the local community centre.

Figure 39: EARL report in PDF format, containing information about Personas

- A new reporting mechanism able to generate reports that contain all the HTTP content transferred between the WaaT and the server that hosts the evaluated web application, using the HTTP Vocabulary in RDF[7], has also been developed within the WaaT. This report can be used as input to other web accessibility evaluators, in order to perform a similar use case scenario and compare their results with those of WaaT. Table 4 presents an example of such a report.

```

<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
  xmlns:http="http://www.w3.org/2011/http#"
  xmlns:dct="http://purl.org/dc/terms/"
  xmlns:cnt="http://www.w3.org/2011/content#">
  <http:Connection rdf:ID="conn0">

  <http:connectionAuthority>www.paypal.com</http:connectio
nAuthority>
    <http:requests rdf:parseType="Collection">
      <http:Request rdf:about="#req0"/>
    </http:requests>
  </http:Connection>
  <http:Request rdf:ID="req0">
    <http:httpVersion>1.1</http:httpVersion>
    <http:methodName>GET</http:methodName>
    <http:mthd
rdf:resource="http://www.w3.org/2011/http-methods#GET"/>
    <http:absolutePath></http:absolutePath>
    <http:headers rdf:parseType="Collection">
      <http:RequestHeader>
<http:fieldName>Host</http:fieldName>
      <http:hdrName
rdf:resource="http://www.w3.org/2011/http-
headers#host"/>

<http:fieldValue>www.paypal.com</http:fieldValue>
      </http:RequestHeader>
    </http:headers>
    <http:resp rdf:resource="#resp0"/>
  </http:Request>
  <http:Response rdf:ID="resp0">
    <http:httpVersion>1.1</http:httpVersion>
    <http:statusCodeValue>200</http:statusCodeValue>
    <http:sc rdf:resource="http://www.w3.org/2011/http-
statusCodes#OK"/>
    <http:reasonPhrase>OK</http:reasonPhrase>
    <http:headers rdf:parseType="Collection">
      <http:GeneralHeader>
        <http:fieldName>Date</http:fieldName>
        <http:hdrName
rdf:resource="http://www.w3.org/2011/http-
headers#date"/>
        <http:fieldValue>Fri, 24 Feb 2012 11:41:50
GMT</http:fieldValue>

```

Table 4: HTTP RDF/XML report - Example

4. Introduction to the Web Services Assessment module

The Web services assessment tool (WSaaT) tries for a first time to fill the accessibility gaps in the field of Web Services (WSs). It builds upon the notion of accessible WSs and the WS accessibility assessment framework (WSaaF) introduced and described in the deliverable D5.2. The WSaaF and its guidelines follow the rationale behind W3C WCAG 2.0-based accessibility standardization of web content. It provides the basis towards building future accessible WSs, a task that can be further facilitated by the use of an appropriate Tool (WSaaT), developed with the aim to provide automatic assessment of services, against guidelines of the proposed framework.

That framework has been elaborated and appropriately extended, so as to support also the emerging field of REST-based WSs. Moreover, the updated version of the tool has been extended towards further facilitating developers of WSs to build either SOAP or REST -based accessible products.

By building upon the notion of an accessible web service, our WS accessibility assessment tool aims to ensure that web services are developed so as to allow for WS - based HCI be accessible both on presentation and content level. This can be done by enhancing WSs with proper accessibility features.

The intent of these features is to ensure that the Client Application - WS interaction part of the WS utilization chain (Figure 40) allows for accessible HCI at the End User - Client Application level. Within WSaaF, WS accessibility is defined on the basis of a three-layer-architecture, comprised of the “core functional”, “basic accessibility” and “extended accessibility” layers. The rationale behind the proposed layers of accessibility is that in order for a service to be considered as fully accessible it has to:

- Be well-defined, well-working and easy to integrate within client applications, so as for developers of client applications to be able to use the service’s functionality and/or provided information effectively within their developed application’s operational context. This requirement defines the concept behind the core functional layer.
- Have accessibility features that will enable the client applications invoking the service to show the delivered content in an accessible way, in respect of the special needs of impaired user groups. This requirement defines the concept behind the basic accessibility layer.
- Provide data which contains enough information, in order for the content itself to be helpful for impaired users, containing information adapted to their special needs. Based on this requirement, the extended accessibility layer is defined.

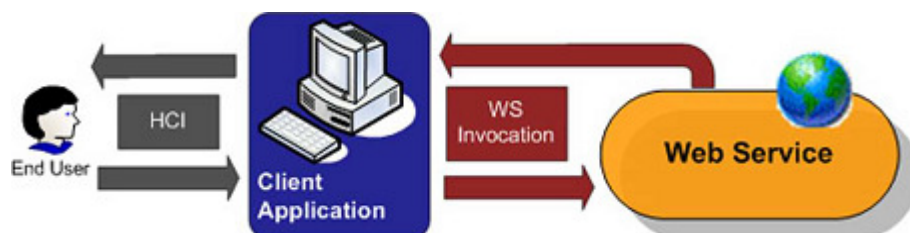


Figure 40: Typical Web Service utilization chain

If regarded as a black box, the WSaaT takes as input the definition of a WS and the WS accessibility guidelines defined within the Accessible Ontology, and produces as output the result of the accessibility assessment process. The service definition can be provided through a “Web Service Description Language” (WSDL) or “Web Application Description Language” (WADL) file, describing the (SOAP or REST -based) WS under evaluation, or even through an appropriate UI that allows users to manually define the specifics of the service. During service assessment, the module communicates with (a) the Accessible Ontology in order to get information regarding the accessibility guidelines defined in the WSaaF and (b) an EARL (Evaluation and Report Language) -based Reporting tool, responsible for translating the assessment result into EARL-based reports. These reports follow a standardized format, and thus can be used for presenting assessment results to the tool users in an appropriate, formalized way.

As depicted in Figure 41, the WSaaT consists of the following main components: the **Web Service Definition Parser**, the **Web Service Definition Editor**, the **Service Alignment Tool** and the **Accessible Web Service Evaluator**.

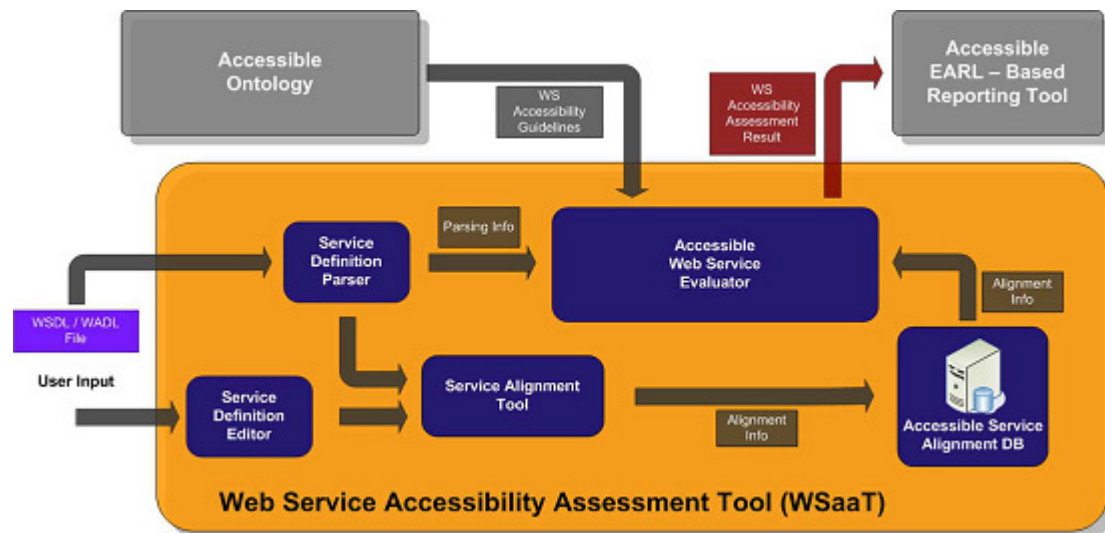


Figure 41: WS Accessibility Assessment Tool (WSaaT) architecture

Web Service Definition Parser

The WS Definition Parser module is responsible for the parsing of formal XML-based files describing a Web Service. The parsing procedure follows a perspective where service operations are regarded as black boxes with documentation, input / output / fault structure elements of either native (e.g. String, boolean, integer, double) or complex types. Complex type elements may consist of children complex or native type elements. In its current implementation, the parsing module supports WSDL and WADL-based service definition files. It takes as input the WSDL or WADL file describing a Web Service (SOAP- or a REST- based respectively) and produces Java structures that hold information regarding the WS's operations, appropriate for further processing and accessibility evaluation. The module is extensible so as to allow (after the integration of further appropriate sub-modules) the parsing of further service types that could be defined in the future (apart from SOAP and REST -based), given that these services' operations could also be regarded as black boxes with specific input and output data structures.

Web Service Definition Editor

Since WADL files are at the present, not commonly used for the description REST WSs, the Tool offers its users the capability to also manually define the structure of their WS through an appropriate UI, without needing to provide any definition file (e.g. WADL) that describes the service. Thus, in this case, the use of the service definition parser can be omitted, and the WS Definition Editor module can be used instead.

Service Alignment Tool

Different types of WSs have different requirements regarding the accessibility features their input/output structures should support. The Service Alignment Tool module is responsible to provide the WSaaS with information regarding the type of service under evaluation, and also regarding whether accessibility-related features are supported from its input/output elements.

For the purposes of this process, a set of accessible “Ideal Operations” has been defined within the Accessible Ontology. Some of them refer to general purpose WSs, such as: Image, Audio, Video or Textual Info Provider operations, whereas others are more specific and deal for instance with “info-mobility” WSs, such as the “points of interest info provider” and “route calculation” operations. The Ideal Operations defined within the Accessible Ontology, utilized by the WSaaS, are listed in Table 5.

The present module offers the service evaluator the capability to “align” the WS operation under assessment to an accessible WS “Ideal” one. The alignment process eventually enables the tool to identify whether specific requirements are met from the input/output structures of the service under assessment, on the basis of the service type it belongs.

Being based on the afore-described WS accessibility guidelines, the ideal operations defined within our framework, hold the minimum necessary input/output elements that can enable the provision of accessible content or information, appropriately tailored to an end user’s special needs. For instance, the image provider ideal operation holds the minimum necessary elements that a WS delivering images (e.g. maps) should have, in order for it to be considered as accessible.

In particular, an alternative text element should accompany every image delivered through such a WS. This alternative text should be a description of the image, capable to be handled for example by appropriate text-to-speech modules integrated in the end user application utilizing the service, so as for the information conveyed through the image to be accessible also to blind end users. Similarly, the “Points of Interest Info Provider” ideal operation defines that a web service of this kind, should provide in each delivered block of information that regards a specific POI (e.g. a restaurant), additional information regarding the POIs accessibility status in respect of different end user categories (e.g. wheelchair user etc.).

As described in more detail in the deliverable D5.2, by semi-automatically aligning the inputs/outputs of their WS operations to the inputs/outputs of the ideal ones during the WS accessibility assessment process, the developers of the WS (users of the tool) can provide the WSaaS with information regarding whether specific elements exist in their development, necessary to make it accessible in respect of the domain it belongs to. Additionally, it should be noted that apart from enabling the tool to identify the WS type and whether accessibility-related elements exist in the input/output data structures, these ideal operations provide developers also with a more direct guide

over accessibility characteristics that their services should have so as for them to be regarded as accessible.

Ideal Operation	Description	Inputs	Outputs
Image Provider	An operation that provides one or more images as output	-	Image Object or Image Object URL, Alternative Text
Audio Provider	An operation that provides one or more audio files as output	-	Audio Object or Audio Object URL, Title, Short Description, Full Transcript
Video Provider	An operation that provides one or more video files as output	-	Video Object or Video Object URL, Title, Short Description, Full Transcript, Audio Description of video {Audio Description Object or Audio Description Object URL, Audio Description Title, Audio Description Short Description, Audio Description Full Transcript, Audio Description Sign Language Interpretation}
Textual Info Provider	An operation that receives input or provides as output information in the form of text	Desired Language	Text Object or Text Object URL, Image Of Text, Text Language, Textual Info In English, Link To Auditory Or Graphic Presentation, Simple Text for Secondary Education Level
"Points of Interest" Info Provider	An operation that provides information about "Points of Interest" as output	User Group	Accessible Point of Interest: POI general info, User Group, POI Accessibility Status, POI Accessibility Status Details
"Route calculation" Operation	An operation that provides route calculation capabilities and produces "calculated routes" as output	User Group	Accessible Route: a. Pedestrian Route Segment{RS general info, User group, Accessibility Status, Accessibility Status details} b. Public Transport Route Segment{ RS general info, Origin Accessibility Status{User group, Accessibility Status, Accessibility Status details} Destination Accessibility Status{User group, Accessibility Status, Accessibility Status details} Means (Vehicle){User Group, Accessibility Status, Accessibility Status details}}

Table 5: The Ideal Operations defined within the Accessible Ontology, utilized by the WSaaS

Accessible Web Service Evaluator

The Accessible Web Service Evaluator is responsible for interpreting and combining information gathered from the above-described modules, so as to conclude whether the service under assessment belongs to a specific accessibility class or not. In particular, it takes as input (a) the information derived from the parsing of the WSDL/WADL file, or the manual editing of the WS definition (through the WS Definition Editor module), (b) the information derived from the alignment of the WS operation to the concepts defined within the "Ideal" ones and (c) the WS accessibility guidelines defined within the WSaaSF. The module combines these three inputs and produces as output the WS accessibility assessment result, which is then passed to the Accessible EARL based reporting tool, responsible for the EARL-based Accessibility Report generation.

4.1. Web Services Accessibility Assessment (using the ACCESSIBLE Portal)

The Web services Accessibility Assessment Tool can be accessed by clicking on the "Web Services Assessment Projects" through the ACCESSIBLE portal. The assessment procedure is similar with the assessment procedure of the Web Accessibility Assessment module. As it is obvious in Figure 42, the user can either add a new project or he/she can assess one of the already added projects (Figure 43).



Figure 42: Web Services Assessment Projects

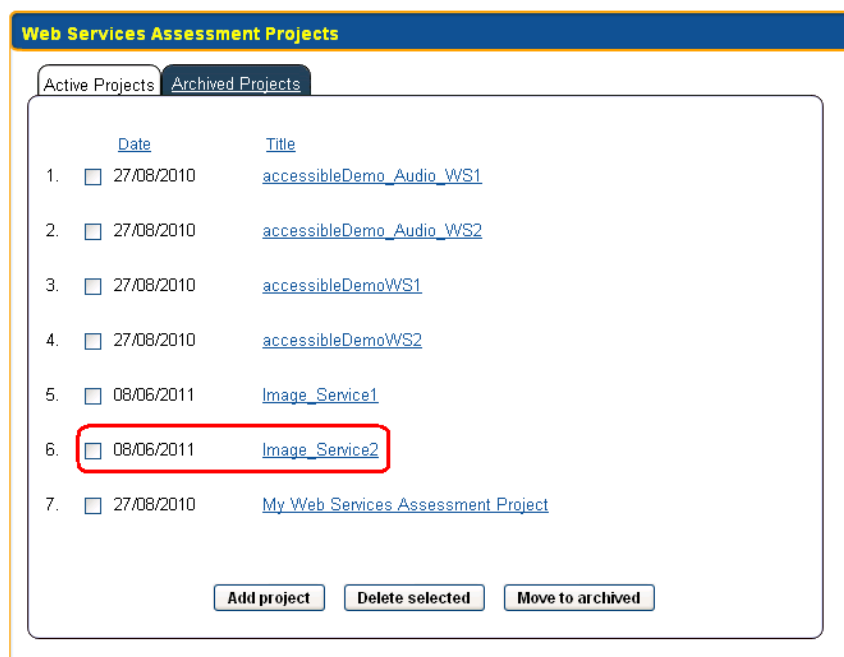


Figure 43: Web Services Assessment Projects

The user through the tool can have the following three different options (Figure 44) to assess the preferred web service:

- *Evaluate your Web Service Descriptions against Personas*
- *Evaluate a collection of Web Services against ontology*
- *Evaluate a collection of Web Services against specific guidelines*



Figure 44: Three options for the evaluation of the Web Services

Evaluating Web Service Descriptions against Personas

By selecting the first option, the user chooses to evaluate its preferable Web Services against Personas. The first step of the procedure is depicted in Figure 45 where all the needed information about the Personas is presented to the user of the tool. Then, the user can select against which Persona, he/she will perform the evaluation process. All the information about the Personas is also available in PDF format.

Step 1 of 5: Select among a number of predefined Personas

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel

Back to project options

Next

☐ Andy Catteeuw



Andy Catteeuw is 23 years old and suffers from Attention Deficit Hyperactivity Disorder
Marital status: Single Job: Student Location: Houthalen, Belgium Education:
[Download persona description in pdf format](#)

☐ Matthew Perkins



Matthew Perkins is 28 years old and suffers from Blurred Vision, Down Syndrome
Marital status: Single Job: Employed Location: Birmingham, UK Education:
[Download persona description in pdf format](#)

☒ Nitesh Sarin

Nitesh Sarin is 36 years old and suffers from Communication Disability, Learning Disability, Colour Blindness
Marital status: Single Job: Unemployed Location: Pocklington, UK Education: College

☐ Carlos Portillo



Carlos Portillo is 35 years old and suffers from Communication Disability, Expressive Language Disorder
Marital status: Married Job: Employed Location: Gerona, Italy Education: College degree in Applied Informatics
[Download persona description in pdf format](#)

Figure 45: Evaluation against Personas, step 1

In Step 2 a list with the initial set of Guidelines is presented to the user (Figure 46) where he/she can select which approaches could be activated during the evaluation process.

Step 2 of 7: Edit the list of evaluation approaches to be applied

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

The following checkpoints have been inferred from your project settings.

Guideline_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.

- ☒ WebService1 Checkpoint_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.
- ☒ WebService1 Checkpoint_C1.1
Provide the accessibility status of all the Points of Interest returned upon service invocation
- ☒ WebService1 Checkpoint_C1.2
Provide a description of the accessibility status of all the Points of Interest returned upon service invocation

Guideline_C2
When a service provides content related to Route Guidance, it should provide information describing efficiently and effectively the accessibility status of the various route segments, regarding a set of different impaired user groups with different needs.

- ☒ WebService1 Checkpoint_C2
When a service provides content related to Route Guidance, it should provide information describing efficiently and effectively the accessibility status of the various route segments, regarding a set of different impaired user groups with different needs.
- ☒ WebService1 Checkpoint_C2.1
For each pedestrian segment of any Route returned upon service invocation, provide its accessibility status regarding various groups of impaired users
- ☒ WebService1 Checkpoint_C2.2
For each pedestrian segment of any Route returned upon service invocation, provide a description of its accessibility status regarding various groups of impaired users. This description must contain for all the different user groups enough information in order to enable them move along the specified route, in respect to their special needs

Figure 46: Evaluation against Personas, step 2

The next step is the selection of the operation that will be evaluated, as shown in Figure 47. The available options are presented to the user and he/she can choose the Operation that wishes to evaluate.

Step 3 of 7: Select Operation to Evaluate

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

☒ Operation: getMap_NEW

Inputs

-
-

Outputs

-
-
-
-
-
-

Figure 47: Evaluation against Personas, step 3

Step 4 includes the results of the evaluation process before the alignment procedure, which are available in four different formats:

- *Full Report*
- *Report only errors*
- *EARL report (xml)*
- *EARL report (pdf)*

Figure 48 depicts the *full report*, whereas Figure 49 and Figure 50 present the EARL report in xml and pdf format respectively.

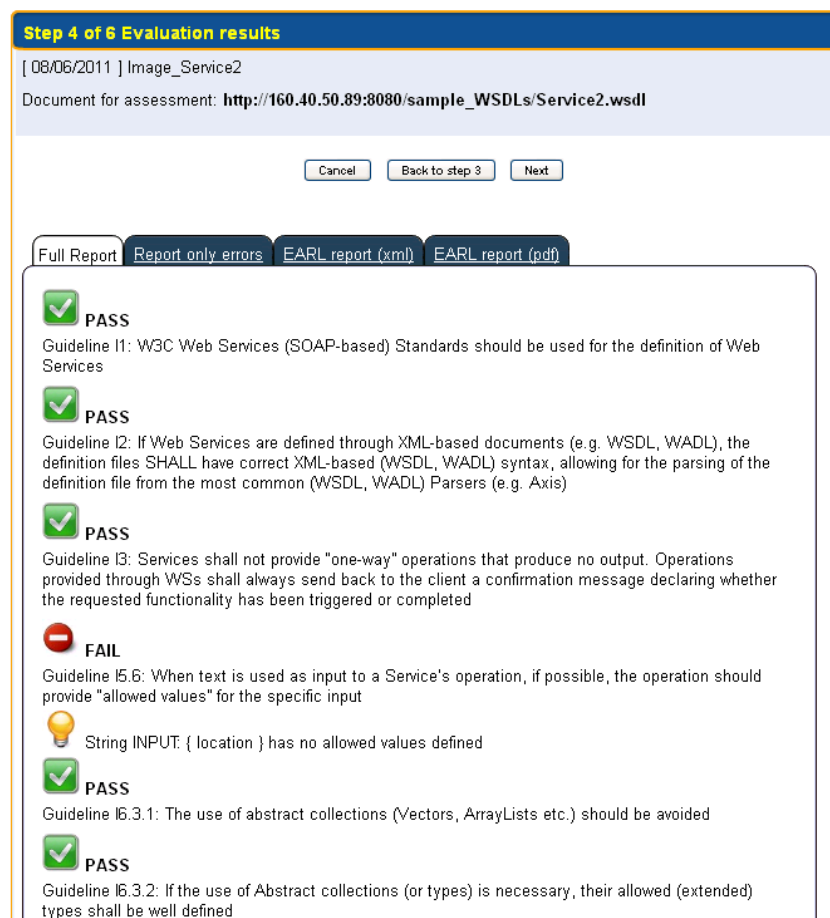


Figure 48: Full report



Figure 49: EARL report (xml format)

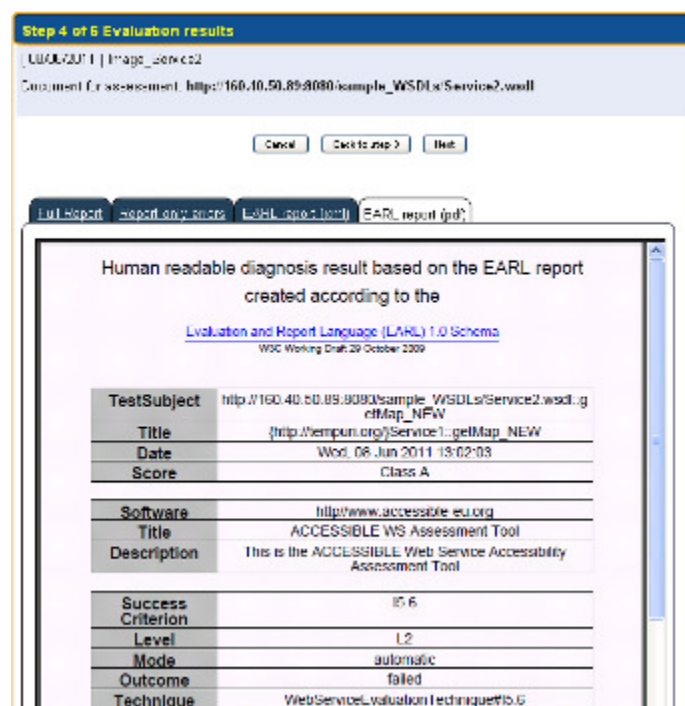


Figure 50: EARL report (pdf format)

By pressing the button “Next” the user proceeds to step 5, where as shown in Figure 51, the user has to choose one of the ACCESSIBLE Ideal Operations that better describe the Service’s Operation effectively. If no Ideal Operation that describes your

Operation effectively is found, no Ideal Operation should be selected at this point. “Operation Alignment” is the process during which an Operation of a WS is declared to be of a specific type, by aligning it to one of the “Ideal Operations” (see Annex 10.1) defined within the Tool. During this process, the user selects an “Ideal Operation” that s/he believes is more similar to the Operation under evaluation, and then the inputs and outputs of the Operation are semi-automatically aligned to corresponding ones that belong to the selected “Ideal Operation”.

Step 5 of 6 Align with ACCESSIBLE's Ideal Operations

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 4 Next

Service supported operations

Operation: getMap_NEW

Inputs

- location{http://www.w3.org/2001/XMLSchema:string}
- useLocalhost{http://www.w3.org/2001/XMLSchema:boolean}

Outputs

- getMap_NEWResult{http://tempuri.org/ITI_Image_NEW}
 - imageUrl{http://www.w3.org/2001/XMLSchema:string}
 - imageType{http://www.w3.org/2001/XMLSchema:string}
 - imageAlternativeText{http://www.w3.org/2001/XMLSchema:string}
 - imageWidthInPixels{http://www.w3.org/2001/XMLSchema:int}
 - imageHeightInPixels{http://www.w3.org/2001/XMLSchema:int}

ACCESSIBLE ideal operations

☒ Operation: ImageProvider

Inputs

Outputs

- AccessibleImage{ImageType}
 - ImageObjectURL{String}
 - AlternativeText{String}
 - ImageObject{anyType}

Figure 51: Evaluation against Personas, step 5

If an Ideal Operation has been selected, an Automatic input and output Matching Process takes place (Figure 52). This process uses a Web Service that computes similarity matching scores between the names of your Operation’s input and output elements and the names of the selected Ideal Operation’s input and output elements. Using this matching scores calculation Web Service, the Tool tries to match and align the inputs and outputs of your operation with the ones belonging to the Ideal Operation.

The final results of the evaluation process are presented to the user, as shown in Figure 53. The evaluation outcome for each Guideline is either “PASS” or “FAIL”. For the Guidelines that FAIL the evaluation, a brief description of the Failure reason is provided.

Step 6 of 7 Align Operation

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 5 Next

Web Service Operation

- ☐ location{http://www.w3.org/2001/XMLSchema:string}
- ☐ useLocalhost{http://www.w3.org/2001/XMLSchema:boolean}
- ☒ getMapResult{http://tempuri.org::ITI_Image}
- ☒ imageUrl{http://www.w3.org/2001/XMLSchema:string}

ACCESSIBLE Ideal Operation

- ☒ AccessibleImage{ImageType}
 - ☒ ImageObjectURL{String}
 - ☐ AlternativeText{String}
 - ☐ ImageObject{anyType}

Align Selections

Figure 52: Evaluation against Personas, step 6

Step 7 of 7: Aligned Operation Assessment Results

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 6

Full Report Report only errors EARL report (xml) EARL report (pdf)

☒ **PASS**

Guideline I1: W3C Web Services (SOAP-based) Standards should be used for the definition of Web Services

☒ **PASS**

Guideline I2: If Web Services are defined through XML-based documents (e.g. WSDL, WADL), the definition files SHALL have correct XML-based (WSDL, WADL) syntax, allowing for the parsing of the definition file from the most common (WSDL, WADL) Parsers (e.g. Axis)

☒ **PASS**

Guideline I3: Services shall not provide "one-way" operations that produce no output. Operations provided through Ws shall always send back to the client a confirmation message declaring whether the requested functionality has been triggered or completed

☐ **FAIL**

Guideline I5.6: When text is used as input to a Service's operation, if possible, the operation should provide "allowed values" for the specific input

☐ String INPUT: { location } has no allowed values defined

☒ **PASS**

Guideline I6.3.1: The use of abstract collections (Vectors, ArrayLists etc.) should be avoided

Figure 53: Evaluation against Personas, step 7

Evaluating a collection of Web Services against ontology

The second option is to evaluate a collection of Web Services against Ontology. More specifically to evaluate this collection of Web Services using the Harmonized methodology, which is described in D3.1 “ACCESSIBLE harmonizes methodology”. By clicking on the button “*Evaluate a collection of Web Services against Ontology*”, the Disabilities are presented to the user. Then, the user has the opportunity to choose between the Disabilities that are included in the Harmonized methodology. This procedure is depicted in following Figure 54.

Step 1 of 7: Select among a number of evaluation categories for the ontology

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to project options Next

Disability Functional Limitation Impairment ATs Software

- ☐ Night blindness
- ☐ Deafness
- ☐ Cerebral palsy
- ☐ Deaf-Blindness
- ☒ Arthritis
- ☐ Asperger's syndrome
- ☐ Quadriplegia
- ☐ Dyspraxia
- ☐ Speech and language disorders

Figure 54: Evaluation against ontology, step 1

The next step, namely Step 2, presents to the user the Web Service Guidelines, which correspond to the Disability that was selected in the first step. Thus, the user can review and edit these guidelines concerning the evaluation process that will follow.

Step 2 of 7: Edit the list of evaluation approaches to be applied

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 1 Next

The following checkpoints have been inferred from your project settings.

Guideline_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.

- ☒ WebService1 Checkpoint_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.
- ☒ WebService1 Checkpoint_C1.1
Provide the accessibility status of all the Points of Interest returned upon service invocation
- ☐ WebService1 Checkpoint_C1.2
Provide a description of the accessibility status of all the Points of Interest returned upon service invocation

Guideline_C2
When a service provides content related to Route Guidance, it should provide information describing efficiently and effectively the accessibility status of the various route segments, regarding a set of different impaired user groups with different needs.

- ☒ WebService1 Checkpoint_C2
When a service provides content related to Route Guidance, it should provide information describing efficiently and effectively the accessibility status of the various route segments, regarding a set of different impaired user groups with different needs.

Figure 55: Evaluation against ontology, step 2

The following steps of the evaluation procedure are the same as the ones described in *Evaluating your Web Service Descriptions against Personas*. In Step 3, the user selects the Operation to be evaluated (Figure 56).

Step 3 of 7: Select Operation to Evaluate

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 2 Next

☒ Operation: getMap_NEW

Inputs

- location{http://www.w3.org/2001/XMLSchema:string}
- useLocalhost{http://www.w3.org/2001/XMLSchema:boolean}

Outputs

- getMap_NEWResult{http://tempuri.org:ITI_Image_NEW}
 - imageURL{http://www.w3.org/2001/XMLSchema:string}
 - imageType{http://www.w3.org/2001/XMLSchema:string}
 - imageAlternativeText{http://www.w3.org/2001/XMLSchema:string}
 - imageWidthInPixels{http://www.w3.org/2001/XMLSchema:int}
 - imageHeightInPixels{http://www.w3.org/2001/XMLSchema:int}

Figure 56: Evaluation against ontology, step 3

Step 4 of 6 Evaluation results

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 3 Next

Full Report Report only errors EARL report (xml) EARL report (pdf)

 **PASS**

Guideline I1: W3C Web Services (SOAP-based) Standards should be used for the definition of Web Services

 **PASS**

Guideline I2: If Web Services are defined through XML-based documents (e.g. WSDL, WADL), the definition files SHALL have correct XML-based (WSDL, WADL) syntax, allowing for the parsing of the definition file from the most common (WSDL, WADL) Parsers (e.g. Axis)

 **PASS**

Guideline I3: Services shall not provide "one-way" operations that produce no output. Operations provided through WSSs shall always send back to the client a confirmation message declaring whether the requested functionality has been triggered or completed

 **FAIL**

Guideline I5.6: When text is used as input to a Service's operation, if possible, the operation should provide "allowed values" for the specific input

 String INPUT: { location } has no allowed values defined

Figure 57: Evaluation against ontology, step 4

The first evaluation results are presented to the user, as shown in Figure 57, whereas steps 5 and 6 include the alignment process steps (Figure 58 and Figure 59). Finally, the results after the alignment process are presented to the user (Figure 60).

Step 5 of 6 Align with ACCESSIBLE's Ideal Operations

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 4 Next

Service supported operations

Operation: getMap_NEW

Inputs

- location{http://www.w3.org/2001/XMLSchema:string}
- useLocalhost{http://www.w3.org/2001/XMLSchema:boolean}

Outputs

- getMap_NEWResult{http://tempuri.org/ITI_Image_NEW}
- imageUrl{http://www.w3.org/2001/XMLSchema:string}
- imageType{http://www.w3.org/2001/XMLSchema:string}
- imageAlternativeText{http://www.w3.org/2001/XMLSchema:string}
- imageWidthInPixels{http://www.w3.org/2001/XMLSchema:int}
- imageHeightInPixels{http://www.w3.org/2001/XMLSchema:int}

ACCESSIBLE ideal operations

☒ Operation: ImageProvider

Inputs

Outputs

- AccessibleImage{ImageType}
- ImageObjectURL{String}
- AlternativeText{String}
- ImageObject{anyType}

Figure 58: Evaluation against ontology, step 5

Step 6 of 7 Align Operation

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 5 Next

Web Service Operation

- location{http://www.w3.org/2001/XMLSchema:string}
- useLocalhost{http://www.w3.org/2001/XMLSchema:boolean}
- getMapResult{http://tempuri.org/:ITI Image}
- imageURL{http://www.w3.org/2001/XMLSchema:string} (aligned to: ImageObjectURL)**

ACCESSIBLE Ideal Operation

- AccessibleImage{ImageType}
- ImageObjectURL{String}
- AlternativeText{String}
- ImageObject{anyType}

Align Selections

Figure 59: Evaluation against ontology, step 6

Step 7 of 7: Aligned Operation Assessment Results

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Cancel Back to step 6

Full Report Report only errors EARL report (xml) EARL report (pdf)

FAIL

Guideline I5.6: When text is used as input to a Service's operation, if possible, the operation should provide "allowed values" for the specific input

String INPUT: { location } has no allowed values defined

FAIL

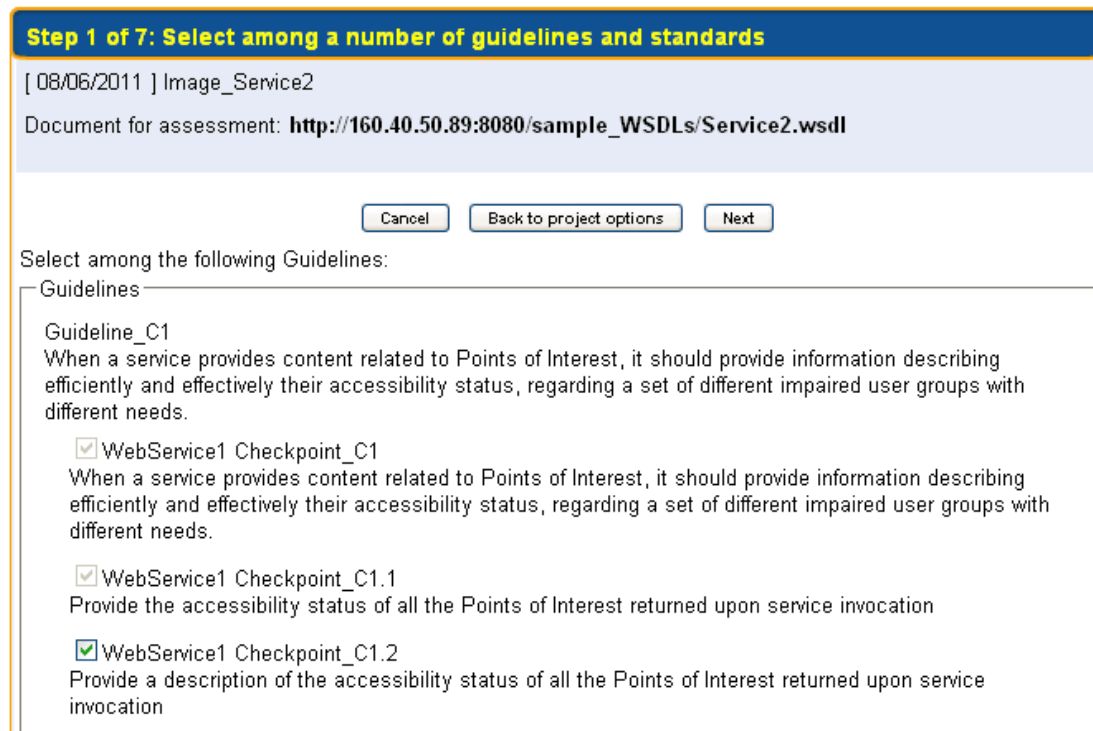
Guideline I6.5.1: Provide comprehensible error messages, that hold adequate information, within "Fault Message structures" explicitly defined at the operations' output

Operation getMap has no output Fault Message element defined in its structure

Figure 60: Evaluation against ontology, step 7

Evaluating a collection of Web Services against specific guidelines

The final option is to evaluate a collection of Web Services against specific guidelines as described in the Web services framework. Thus, a number of Guidelines were developed and are available in the D3.1 "ACCESSIBLE harmonizes methodology". Thus, in the first step, all the Web Services guidelines are presented to the user and he/she can select the guidelines that he/she wishes to be used during the evaluation procedure (Figure 61). The remaining steps are the same as described before in the *Evaluating a collection of Web Services against specific guidelines*.



Step 1 of 7: Select among a number of guidelines and standards

[08/06/2011] Image_Service2

Document for assessment: http://160.40.50.89:8080/sample_WSDLs/Service2.wsdl

Select among the following Guidelines:

Guidelines

Guideline_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.

☐ WebService1 Checkpoint_C1
When a service provides content related to Points of Interest, it should provide information describing efficiently and effectively their accessibility status, regarding a set of different impaired user groups with different needs.

☐ WebService1 Checkpoint_C1.1
Provide the accessibility status of all the Points of Interest returned upon service invocation

☒ WebService1 Checkpoint_C1.2
Provide a description of the accessibility status of all the Points of Interest returned upon service invocation

Figure 61: Evaluation against specific guidelines, step 1

4.2. Web Services Accessibility Assessment (using the standalone tool)

In the present section, a typical use case scenario by using the developed standalone Tool is being described. For this purpose, an example WS is taken into account, which takes as input the name of a city and provides as output an image showing a corresponding map. From the process described in the following, the service is assessed against a subset of the defined accessibility guidelines, whose automatic assessment is feasible. Of course, the evaluator (e.g. the developer of the WS) may also manually check whether her/his service conforms to all of the guidelines defined within the framework. In any case, the use of the tool can significantly facilitate the process of WS accessibility assessment.

As described in the previous sections, the assessment procedure supported from the WS assessment Tool consists of the following steps:

Step 1. Providing information over the WS operations' structure

The goal of this initial step is to provide the Tool with information regarding the operations defined within the service under evaluation. This can be done in two ways, in respect of whether the developer possesses a definition file describing the service, or not, as described in the following.

If a WSDL or WADL file describing the SOAP or REST WS exists, the capabilities of the Tool's Service Definition Parser module can be utilized, so as for information regarding the structure of the service to be automatically acquired. In this case, the user is offered with two options, either to browse for a service definition file located within the PC's file system, or to provide a URL pointing to the web address where the definition file can be found. In both these cases, the Parser module takes as input

the service definition file (WSDL or WADL), parses it, and provides as output appropriate Java-based structures that contain all relevant information extracted (Figure 62). This information can thereafter be processed by the Tool's Accessible Web Service Evaluator module, towards WS accessibility assessment.

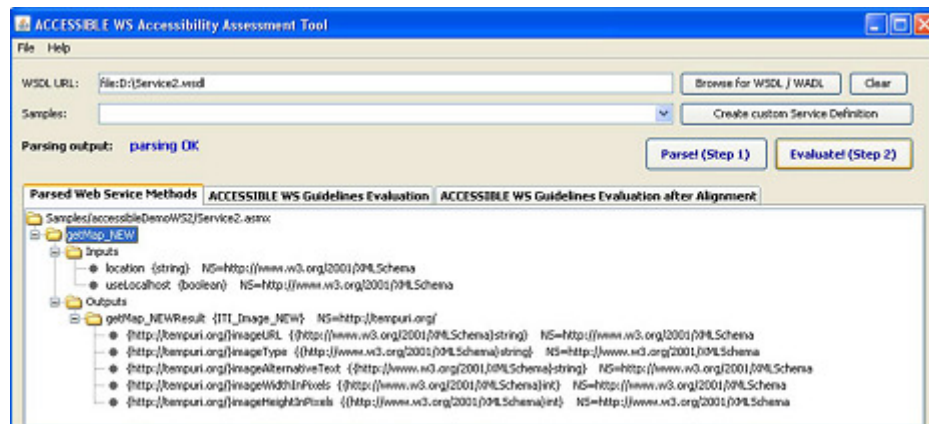


Figure 62: WS accessibility assessment Step 1: Parsing of service definition file

There could be cases however, where a developer would wish to evaluate the accessibility of a service that is not accompanied by a respective definition file. For instance, as said, REST services are rarely accompanied by a WADL file describing them. Our developed Tool can facilitate the user also in this case. In particular, the “Create Custom Definition” option (Figure 62) presents the user interface of the Service Definition Editor module (Figure 63). Using this module, the developer can manually define the specifics of the service s/he wishes to evaluate. The user can insert optional information regarding e.g. the service name and, more importantly, to define the operations the service consists of (“Add New Operation” option).

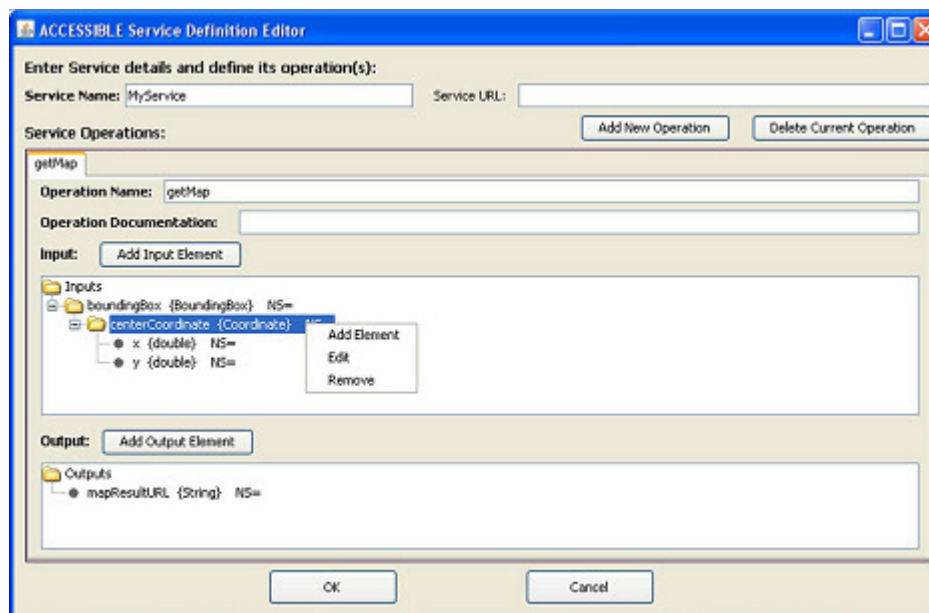


Figure 63: WS accessibility assessment Step 1: The Service Definition Editor

Through the last option, the developer is initially asked to provide the name of the operation. After that, the input and output elements of the operation can be set (Figure 64).

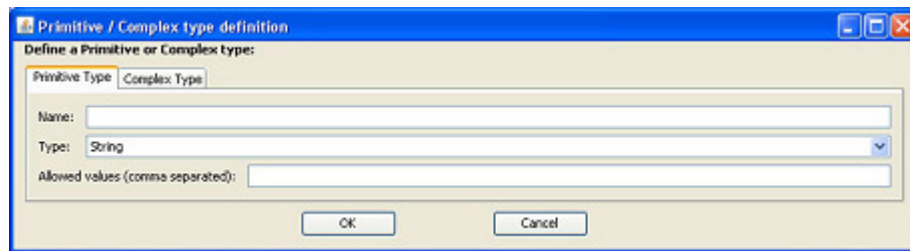


Figure 64: Manual definition of a WS operation input/output elements

Herein, the user can define either a primitive (by providing its name, type and optionally allowed values) or a complex (by providing its name) type of input/output. In the case of a complex type element, the user can subsequently define the elements it consists of, by following exactly the same procedure described above. The user has the option to also insert the documentation of each defined operation. With the afore-described procedure, the user can manually provide the tool with the definition of the service s/he wishes to evaluate. The output of the Service Definition Editor module is similar to the output of the Service Definition Parser, thus the result can thereafter be similarly processed from the Accessible Web Service Evaluator.

Step 2. Automatic evaluation of the WS accessibility status based on information provided through Step 1

Within this step, through the option “Evaluate (Step 2)” shown in Figure 65, all information acquired from step 1 is used from the Web Service Evaluator in order to assess the given operation against a limited set of guidelines. Indicative results of this process are shown in Figure 65. This limited set of guidelines contains those, which can be automatically checked by using only the information acquired so far from step 1. In their majority, these are Level 1 guidelines referring to core functional accessibility features that a WS should have, ensuring its potential for proper integration within service consuming applications.

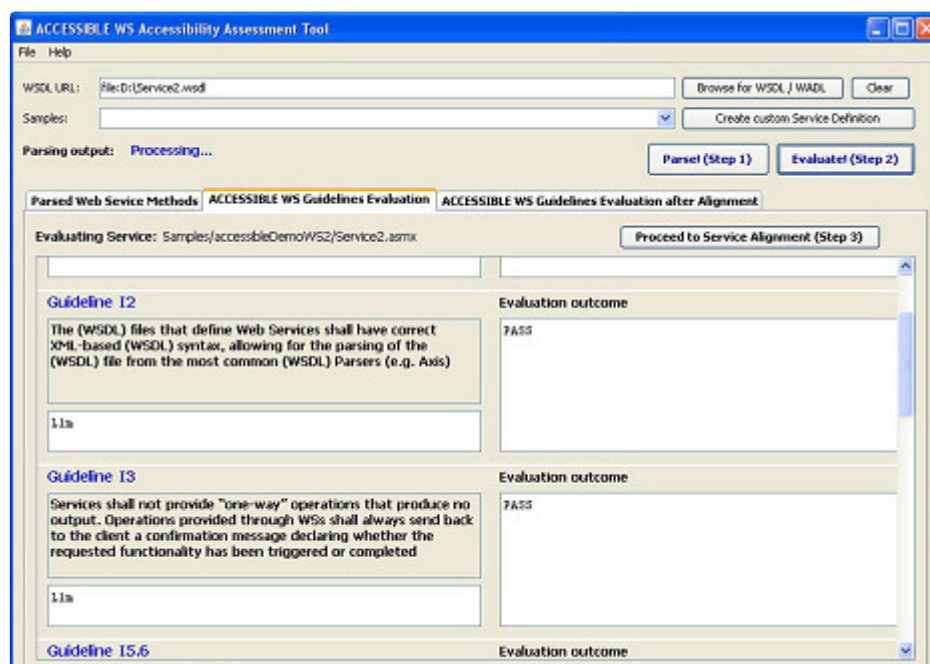


Figure 65: WS accessibility assessment Step 2: Automatic assessment of initial subset of guidelines

In respect of WSDL and WADL files describing WSs, accessibility assessment can up to this step be conducted fully automatically. Thus, the assessment against this subset of guidelines can even be conducted in a batch-job fashion, over large numbers of WSs and their operations. This functionality is also supported by the Tool, taking as input a local folder containing WSDL or WADL files, and providing as output EARL-based reports, each describing the assessment result for each WS operation. Nevertheless, the next step of the overall assessment procedure (service alignment) requires further interaction from the Tool's user, as described in the following.

Step 3. Alignment of the service's operations to the Accessible "Ideal Operation" elements, defined within the Ontology

By utilizing the service alignment capabilities offered from the Service Alignment Tool, the WSaaT acquires in this step further information regarding the Service's operations and their input and output structures. Within this process, the user first selects the Ideal operation that the operation under evaluation resembles to (Figure 66). Once the preferred Ideal operation is selected, the user is presented with the UI that enables her/him to align the input/output elements of the two operations (the Ideal and the one under assessment) (Figure 67).

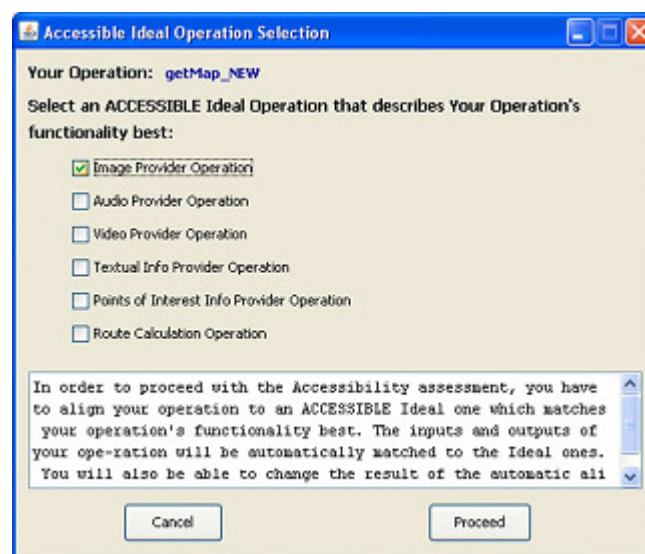


Figure 66: WS accessibility assessment Step 3: Selection of operation type, according to the accessible ideal operations

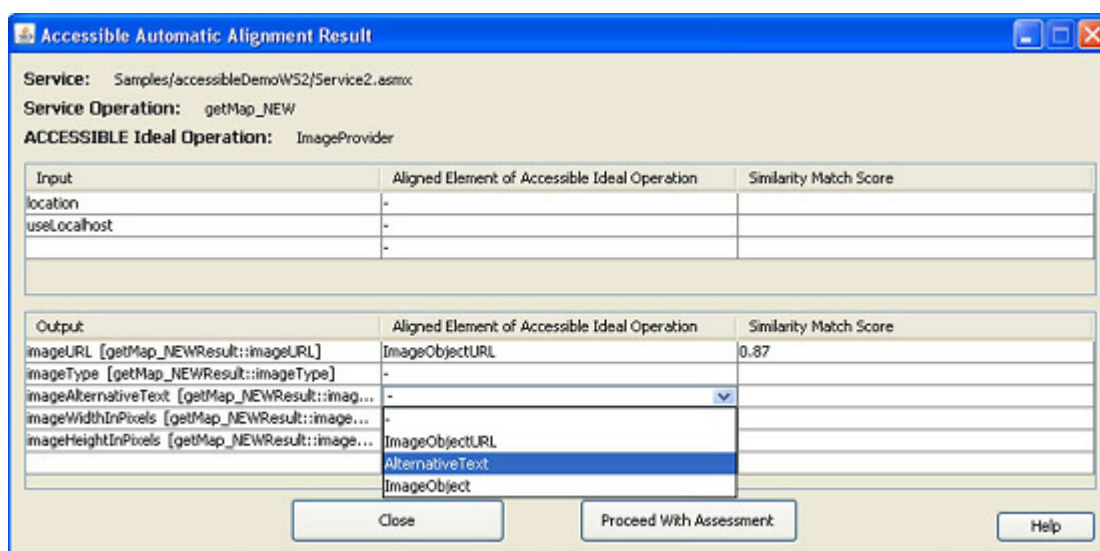


Figure 67: WS accessibility assessment Step 3: Alignment of operation's inputs/outputs to respective ones of the ideal operation

The alignment process is facilitated at this point through the automatic identification of input/output elements of the two operations, that could be potentially aligned. This is done through the calculation of similarity scores among the names of all elements. In case a significantly high score is obtained for a specific pair of elements, the tool provides the user with a recommendation regarding this fact. In particular, in case the similarity matching score is over the empirically defined threshold of 0.8, this fact is depicted in the UI that enables the user align the inputs/outputs of the operations. Once the service alignment is complete, the alignments produced are ready to be used from the Accessible WS Evaluator module, during the following, final step of the assessment.

Step 4. Automatic evaluation of the Service's accessibility status based on information acquired from steps 1 and 3

Within this step, all information acquired from Steps 1 and 3 is used in order for the WSaaS to evaluate whether the service conforms to a broader set of Guidelines than the one assessed during Step 2. The user is finally provided with a cumulative assessment result, taken from the evaluation of the WS against all Level 1, Level 2 and Level 3 guidelines that could be automatically assessed, on the basis of all information acquired so far. In the example depicted from Figure 68, the operation under assessment was an image provider one, which contained in its outputs, an element used for the purpose of providing alternative text along with the image returned (element `imageAlternativeText`, as shown in Figure 67). As a result, the specific service was found to be a Class AA one (Figure 68), which, as said before, is the optimal class for a "general-purpose" WS.

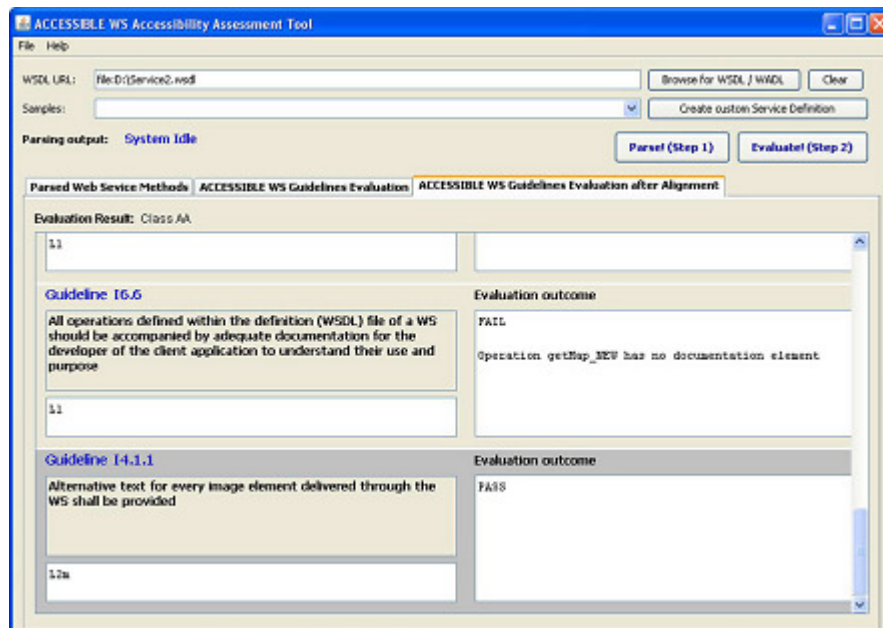


Figure 68: WS accessibility assessment Step 4: Assessment of final set of guidelines

4.3. Changes in current version

Comparing the latest version of Web services Accessibility Assessment module with the previous one, a number of changes were made. These changes are based mainly on the comments received during the review of the project as well as the pilot results phase 2 and 3. Thus some of the following important changes were performed:

- Further updates to the parser of the tool in order to support both SOAP and REST-based services accessibility assessment. So the parser was deeply changed to offer better performance.
- All the bugs found in the pilot tests and further unitary tests were corrected
- Further success/error information is now provided facilitation the accuracy of reporting while appropriate tips and specific help guidelines were integrated to the tool.

5. Mobile Web Accessibility Assessment module

The Mobile Web Accessibility Assessment Tool is the tool developed with goal of assessing web content against both WCAG2 checkpoints and MWBP tests. The assessment is performed simulating the:

- Impairments
- Personas
- Mobile devices used to access and use the content

In order to have an accessibility environment that takes into account the main matters that set back the accessibility to web content through mobile devices.

As the example below (Figure 69) shows, a mobile web content designer could verify the accessibility of her/his website by simulating the access to the content via an iphone used by a blind person.

The tool mitigates the lack of knowledge about accessibility by simulating the impairment and device environment, reduces the necessary experience to develop better accessible web content, and eliminates the heterogeneity of the guidelines and best practices about accessibility by merging MWBP and WCAG2 into a single assessment tool.

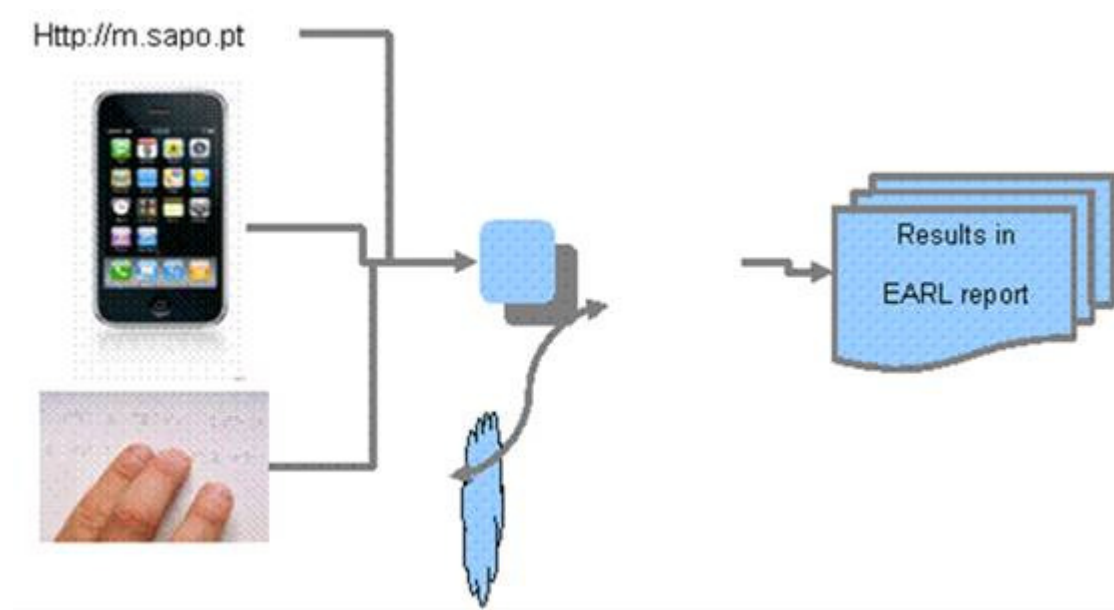


Figure 69: Example of MWAAT

5.1. Mobile Web Accessibility Assessment (using the tool from the ACCESSIBLE portal)

The main page of the ACCESSIBLE portal is depicted in Figure 70. The Web Accessibility Assessment Tool can be accessed by clicking on the “*Mobile Web Applications Assessment Projects*”.

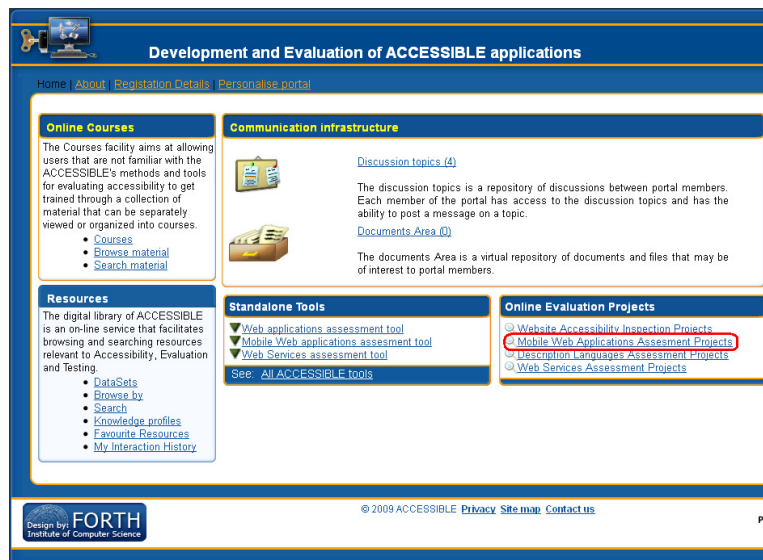


Figure 70: Main view of the ACCESSIBLE portal (Mobile Web Applications Assessment Projects)

In the next step, the active projects are available. The user can either assess an already added project, or he/she can add a new project (Figure 71).

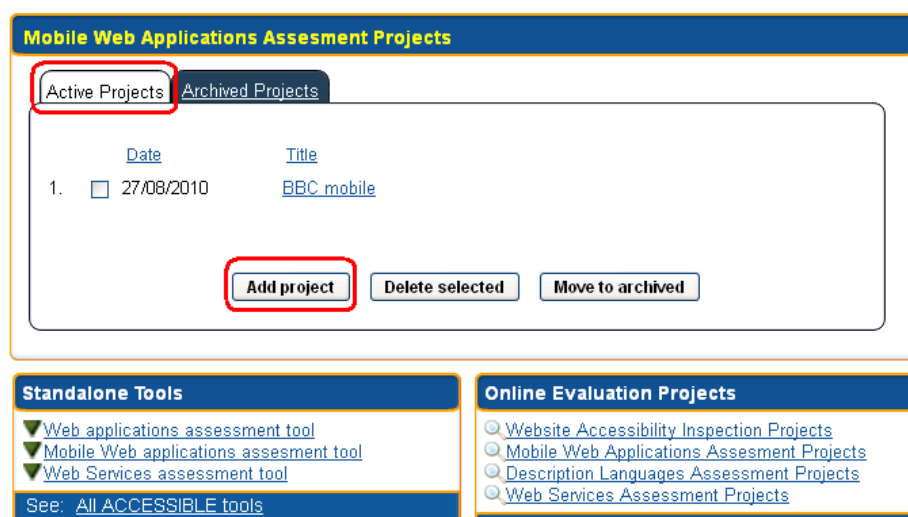


Figure 71: Selection of project to be assessed

If the user decides to add a new project, he/she has to follow the same procedure, as described for the Web Accessibility Assessment tool.

When a user decides to assess a project that is already added, he/she has to select the project he/she wishes to assess and then Figure 72 appears.

Select evaluation method

[27/08/2010] **BBC mobile**
Active document: **index.html**

☒ <http://www.bbc.co.uk/mobile/index.html>
☐ <http://www.bbc.co.uk/news/mobile/world-asia-pacific-11055015>
☐ <http://www.bbc.co.uk/news/mobile/world-asia-pacific-11067471>

Make active

Back to Projects


[Evaluate your Web Site against personas](#)


[Evaluate you mobile web site against ontology](#)


[Evaluate you mobile web site against specific guidelines](#)


[Quick evaluation against ontology](#)

Standalone Tools

- ▼ [Web applications assessment tool](#)
- ▼ [Mobile Web applications assesment tool](#)
- ▼ [Web Services assessment tool](#)

See: [All ACCESSIBLE tools](#)

Online Evaluation Projects

- [Website Accessibility Inspection Projects](#)
- [Mobile Web Applications Assesment Projects](#)
- [Description Languages Assessment Projects](#)
- [Web Services Assessment Projects](#)

Figure 72: Four options for the evaluation of the Web Services

The user has four different options to assess the preferred web site. These options are the same as in the standalone version of the Web Accessibility Assessment Tool and are the following:

- *Evaluate your Web Site against Personas*
- *Evaluate your mobile Web Site against Ontology*
- *Evaluate your mobile Web Site against specific guidelines*
- *Quick evaluation against ontology*

Evaluating a Mobile Web Site against Personas

By clicking in the first option, the user selects to evaluate the Web Site against Personas. The first step of the procedure is depicted in following Figure 73. All the needed information about the Personas is presented to the user. Then, the user can select against which Persona, he/she wants to perform the evaluation process. All the information about the Personas is also available in PDF format.

Step 1 of 3: Select among a number of predefined Personas

[27/08/2010] BBC mobile
Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

☒ Andy Catteeuw

 Andy Catteeuw is 23 years old and suffers from Attention Deficit Hyperactivity Disorder
 Marital status: Single Job: Student Location: Houthalen, Belgium Education: [Download persona description in pdf format](#)

☐ Matthew Perkins

 Matthew Perkins is 28 years old and suffers from Blurred Vision, Down Syndrome
 Marital status: Single Job: Employed Location: Birmingham, UK Education: [Download persona description in pdf format](#)

☐ Nitesh Sarin
 Nitesh Sarin is 36 years old and suffers from Communication Disability, Learning Disability, Colour Blindness
 Marital status: Single Job: Unemployed Location: Pocklington, UK Education: College

☐ Carlos Portillo

 Carlos Portillo is 35 years old and suffers from Communication Disability, Expressive Language Disorder
 Marital status: Married Job: Employed Location: Gerona, Italy Education: College degree in Applied Informatics [Download persona description in pdf format](#)

Figure 73: Evaluation against Personas, step 1

Step 2 includes the review of the corresponding evaluation approaches (Figure 74). The user can select which approaches will be activated during the evaluation process. Moreover, the user can select what kind of device wishes to simulate during the assessment:

- **Mobile:** the content is retrieved by simulating a device representing the characteristics described by the W3C Default Delivery Context
- **Desktop:** the content is retrieved by simulating a desktop computer
- **iPhone:** data retrieved by simulating an iPhone access to the content
- **BlackBerry:** data retrieved by simulating a blackberry access to the content
- **Android:** data retrieved by simulating an android-based device access to the content

By pressing the “Next” button, he/she proceeds to step 3.

Step 2 of 3: Edit the list of evaluation approaches to be applied

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

CancelBack to Step 1Next

Please select one of the following delivery contexts:
☐ Desktop ☒ Mobile ☐ Android ☐ BlackBerry ☐ iPhone

The following evaluation approaches has been inferred for your evaluation setting.
Please, select the ones to be used for the final evaluation:

WCAG2 Evaluation Approaches

☒ Check if the "alt" attribute for elements is empty

☒ Check if there is no "alt" attribute for elements

☒ Check for the existence of "title" attribute for <FRAME> elements

☒ Check for the existence of text equivalent in <OBJECT> elements

☒ Check for the existence of "alt" attribute in <APPLET> elements

☒ Check for the existence of text equivalent in <APPLET> elements

☒ Check for the existence of <TD> elements without "scope" attribute

☒ Check for the existence of <TH> elements without "scope" attribute

☐ Check for the existence of <H1> elements

☐ Check for the existence of <H2> elements

☐ Check for the existence of <H3> elements

☐ Check for the existence of <H4> elements

☐ Check for the existence of <H5> elements

☐ Check for the existence of <H6> elements

Figure 74: Evaluation against Personas, step 2

Figure 75 presents the final step of the evaluation process which includes the Evaluation Report, containing all the errors and the warnings that were detected during the assessment.

Step 3 of 3 Evaluation results

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel Back to step 2

Full Report Report only errors EARL report EARL report (PDF) PDF report Full Report MWBP
EARL report MWBP XML EARL report MWBP (PDF)

WCAG2

Guideline 1.1
Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

WCAG2 SuccessCriterion 1.1.1
Non-text Content: All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.

Check for the existence of "longdesc" attribute in elements

Warnings: 8 (Check for the existence of "longdesc" attribute in elements)

 For the detected elements without "longdesc" attribute, ensure that the "longdesc" attribute is not needed. A "longdesc" attribute should be provided when a short text alternative does not adequately convey the function or information provided in the image

-  Line 21: IMG[attributes={id=webbug, height=1, src=http://www.bbc.co.uk/mobile/webbugproxy...}
-  Line 25: IMG[attributes={src=http://sa.bbc.co.uk/bbc/bbc/s?name=SET-COUNTER&app_type=mobi...}
-  Line 37: IMG[attributes={alt=BBC, src=http://static.bbc.co.uk/mobile/homepage/templates/l...}
-  Line 48: IMG[attributes={alt=, src=http://www.bbc.co.uk/mobile/ic/mobile_4avkeRgn_822f783...}
-  Line 67: IMG[attributes={alt=, src=http://news.bbcimg.co.uk/media/images/53278000/jpg/_53...}
-  Line 136: IMG[attributes={alt=, src=http://news.bbc.co.uk/media/images/53280000/jpg/_53280...}
-  Line 137: IMG[attributes={alt=, src=http://news.bbcimg.co.uk/media/images/53273000/jpg/_53...}

Figure 75: Evaluation against Personas, step 3

The results can be presented to the user with eight different ways. These are:

- *Full Report*
- *Show only Errors/Warnings*
- *EARL report*
- *EARL report (PDF)*
- *PDF report*
- *Full Report MWBP*
- *EARL report MWBP XML*
- *EARL report MWBP (PDF)*

The first five reports correspond to the results provided by the Web Accessibility Assessment tool. As it is already mentioned, the Mobile Accessibility Assessment

tool combines the WCAG2 and MWBP checkpoint into a single assessment tool. Thus, the first five reports concern the Web Accessibility Assessment module and the remaining three concern the detected errors and warnings based on the MWBP checkpoints. Figure 76 depicts the *Full Report MWBP*.

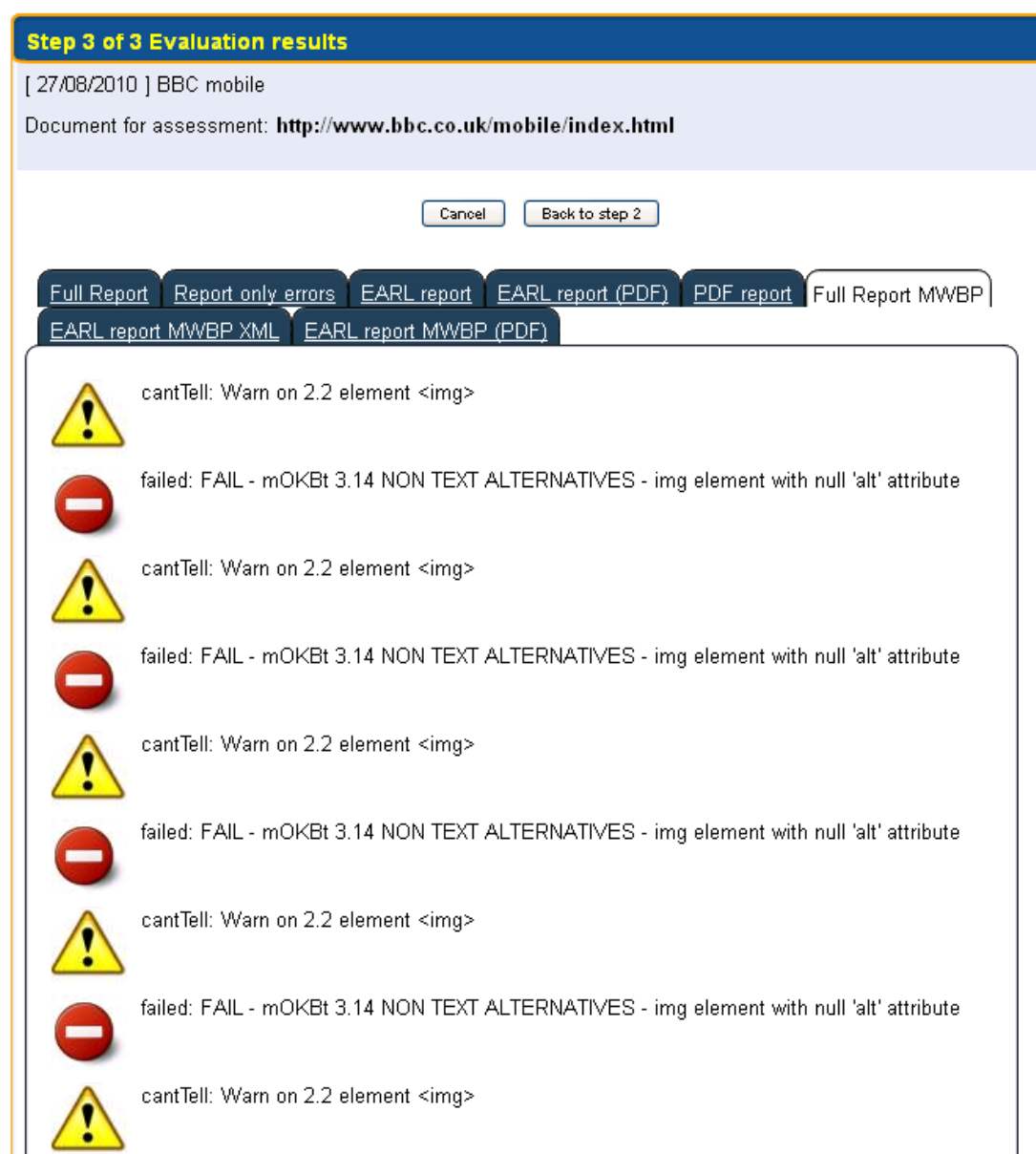


Figure 76: Full report for the MWBP checkpoints

Evaluating a mobile Web Site against Ontology

By clicking in the first option, the user chooses to evaluate the Web Site against the Ontology. More specifically to evaluate the Web Site using the Harmonized methodology, which is described in D3.1 “ACCESSIBLE harmonizes methodology”. By clicking on the button “*Evaluate your mobile Web site against Ontology*”, the Disabilities are presented to the user. Then, the user has the opportunity to choose between the Disabilities (Figure 77) that are included in the Harmonized methodology and then the related (concerning only the selected disability) WCAG2/MWBP tests are automatically selected (Figure 78).

Step 1 of 3: Select among a number of evaluation categories for the ontology

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

- ☒ Night blindness
- ☐ Deafness
- ☐ Cerebral palsy
- ☐ Deaf-Blindness
- ☐ Arthritis
- ☐ Asperger's syndrome
- ☐ Quadriplegia
- ☐ Dyspraxia
- ☐ Speech and language disorders
- ☐ Multiple sclerosis
- ☐ Academic skills disorders
- ☐ Blindness
- ☐ Extreme light sensitivity
- ☐ Attention Deficit Hyperactivity Disorder
- ☐ Mixed Hearing Loss

Figure 77: Evaluation against ontology, step 1

As it is obvious in Figure 78 both the WCAG2 and MWBP approaches are presented to the user. By pressing the button “*Next*” the evaluation process begins and the results are shown to the user, as depicted in Figure 79.

Step 2 of 3: Edit the list of evaluation approaches to be applied

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel

Back to Step 1

Next

Please select one of the following delivery contexts:

☐ Desktop ☐ Mobile ☐ Android ☐ BlackBerry ☒ iPhone

The following evaluation approaches has been inferred for your evaluation setting.

Please, select the ones to be used for the final evaluation:

WCAG2 Evaluation Approaches

- ☒ Check if the "alt" attribute for elements is empty
- ☒ Check if there is no "alt" attribute for elements
- ☒ Check for the existence of "title" attribute for <FRAME> elements
- ☒ Check for the existence of text equivalent in <OBJECT> elements

MWBP Evaluation Approaches

- ☒ AUTO REFRESH and REDIRECTION
- ☒ IMAGE MAPS
- ☒ NO FRAMES
- ☒ NON TEXT ALTERNATIVES
- ☒ OBJECTS or SCRIPT
- ☒ PAGE TITLE
- ☒ POP UPS
- ☒ TABLE ALTERNATIVES
- ☒ TABLES LAYOUT
- ☒ TABLES NESTED
- ☒ Checkpoint 1.4
- ☒ Checkpoint 2.1

Figure 78: Evaluation against ontology, step 2

Step 3 of 3 Evaluation results

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel Back to step 2

Full Report Report only errors EARL report EARL report (PDF) PDF report Full Report MWBP
EARL report MWBP XML EARL report MWBP (PDF)

WCAG2

Guideline 1.1
Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

WCAG2 SuccessCriterion 1.1.1
Non-text Content: All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.

Check for the existence of "longdesc" attribute in elements

Warnings: 8 (Check for the existence of "longdesc" attribute in elements)

 For the detected elements without "longdesc" attribute, ensure that the "longdesc" attribute is not needed. A "longdesc" attribute should be provided when a short text alternative does not adequately convey the function or information provided in the image

-  Line 21: IMG[attributes={id=webbug, height=1, src=http://www.bbc.co.uk/mobile/webbugproxy...]
-  Line 25: IMG[attributes={src=http://sa.bbc.co.uk/bbc/bbc/s?name=SET-COUNTER&app_type=mobi...}]
-  Line 37: IMG[attributes={alt=BBC, src=http://static.bbc.co.uk/mobile/homepage/templates/l...}]
-  Line 48: IMG[attributes={alt=, src=http://www.bbc.co.uk/mobile/ic/mobile_4avkeRgn_822f783...}]
-  Line 67: IMG[attributes={alt=, src=http://news.bbcimg.co.uk/media/images/53278000/jpg/_53...}]

Figure 79: Evaluation against ontology, step 3

Evaluating a mobile Web Site against specific guidelines

The third option is to evaluate a mobile Web Site against specific guidelines. By clicking on the button “*Evaluate your mobile Web site against specific guidelines*”, the Guidelines and the Success Criteria of MobileOK and WCAG2.0 are presented to the user, as depicted in Figure 80 and Figure 81 respectively.

Step 1 of 3: Select among a number of guidelines and standards

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Select among the following Guidelines:

☒ MobileOK ☐ WCAG2 ☐ CSS

Navigation and Links
Because of the limitations in display and of input mechanisms, the possible absence of a pointing device and other constraints of mobile devices, care should be exercised in defining the structure and the navigation model of a Web site.

- ☒ **Access Keys**
Assign access keys to links in navigational menus and frequently accessed functionality.
- ☒ **Auto Refresh**
Do not create periodically auto-refreshing pages, unless you have informed the user and provided a means of stopping it.
- ☒ **Balance**
Take into account the trade-off between having too many links on a page and asking the user to follow too many links to reach what they are looking for.
- ☒ **External resources**
Keep the number of externally linked resources to a minimum.
- ☒ **Image maps**
Do not use image maps unless you know the device supports them effectively.
- ☒ **Link target format**
Note the target file's format unless you know the device supports it.

Figure 80: Evaluation against specific guidelines, MobileOK guidelines

The user can select against which Success Criteria he/she prefers to be used during the evaluation process. After the selection of the Success Criteria, the user presses the button “Next” and then he/she can review the corresponding approaches, as showed in Figure 82.

Step 1 of 3: Select among a number of guidelines and standards

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel

Back to project options

Next

Select among the following Guidelines:

MobileOK

WCAG2

CSS

Guideline 1.1

Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.

☒ WCAG2 SuccessCriterion 1.1.1

Non-text Content: All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.

Guideline 1.2

Provide alternatives for time-based media.

☒ WCAG2 SuccessCriterion 1.2.1

Audio-only and Video-only (Prerecorded)

☒ WCAG2 SuccessCriterion 1.2.2

Captions are provided for all prerecorded audio content in synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

☒ WCAG2 SuccessCriterion 1.2.3

An alternative for time-based media or audio description of the prerecorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labeled as such.

☒ WCAG2 SuccessCriterion 1.2.4

Captions are provided for all live audio content in synchronized media.

Figure 81: Evaluation against specific guidelines, WCAG2.0 guidelines

Step 2 of 3: Edit the list of evaluation approaches to be applied

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel

Back to Step 1

Next

Please select one of the following delivery contexts:

☐ Desktop ☐ Mobile ☐ Android ☐ BlackBerry ☒ iPhone

The following evaluation approaches has been inferred for your evaluation setting.
Please, select the ones to be used for the final evaluation:

WCAG2 Evaluation Approaches

☒ Check if the "alt" attribute for elements is empty
☒ Check if there is no "alt" attribute for elements
☒ Check for the existence of "title" attribute for <FRAME> elements
☒ Check for the existence of text equivalent in <OBJECT> elements

MWBP Evaluation Approaches

☒ AUTO REFRESH and REDIRECTION
☒ IMAGE MAPS
☒ NO FRAMES
☒ NON TEXT ALTERNATIVES
☒ OBJECTS or SCRIPT
☒ PAGE TITLE
☒ POP UPS
☒ TABLE ALTERNATIVES
☒ TABLES LAYOUT
☒ TABLES NESTED
☒ Checkpoint 1.4
☒ Checkpoint 2.1

Figure 82: Evaluation against specific guidelines, step 2

Finally, by pressing the button “*Next*” the evaluation process begins and the results are show in Figure 83.

Step 3 of 3 Evaluation results

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel Back to step 2

Full Report Report only errors EARL report EARL report (PDF) PDF report Full Report MWBP
EARL report MWBP XML EARL report MWBP (PDF)

Check for the existence of "alt" attribute in images used as "submit" ...

Errors: 1 (Check for the existence of "alt" attribute in images used as "submit" buttons (<INPUT type="image">))

 Provide a non-empty "alt" attribute to images used as submit buttons and check that the "alt" attribute indicates the button's function

-  Line 120: INPUT[attributes={type=image, src=http://static.bbc.co.uk/mobile/homepage/templa...

Check for tabbing order among elements (existence of "tabindex" attribute)

Warnings: 36 (Check for tabbing order among elements (existence of "tabindex" attribute))

 For the detected <A> elements without "tabindex" attribute, ensure that the "tabindex" attribute is not needed

-  Line 28: A[attributes={href=http://news.bbc.co.uk/go/promo/click/mobile/englandsrilanka07...
-  Line 31: A[attributes={href=#15; value=[Search]] at line 31
-  Line 33: A[attributes={class=componentHeaderTitle, href=http://news.bbc.co.uk/1/mobile/de...

Figure 83: Evaluation against specific guidelines, step 3

Quick evaluation against ontology

The last option is the quick evaluation against ontology. By pressing the corresponding button, the user can choose between the different evaluation categories, as depicted in Figure 84. The following steps are the same as in the *evaluation of a mobile Web Site against ontology*.

Step 1 of 3: Select Categories

[27/08/2010] BBC mobile

Document for assessment: <http://www.bbc.co.uk/mobile/index.html>

Cancel Back to project options Next

Evaluation Categories

- ☐ Disability
- ☐ Functional Limitation
- ☒ Impairment
 - Standards
 - ☒ Mobile Ok Guidelines
 - ☒ WCAG2 Guidelines
 - ☒ CSS rules
 - Devices
 - ☐ Alternative Keyboards or Switches
 - ☐ Braille Displays
 - Software
 - ☐ Scanning Software
 - ☐ Screen Magnifiers
 - ☐ Screen Readers
 - ☐ Speech Synthesis & Recognition
 - ☐ Text Browsers

Figure 84: Quick evaluation against ontology, step 1

5.2. Changes in current version

Since the last issued version of the tool a number of changes were made that are summarised in the following:

- Services implementations were deeply changed to offer better performance. In particular new libraries have been used (for example, CFX has replaced the groovy SOAP service facilities) that boosted performance and corrected some bugs.
- Further success/error information is now provided facilitation the accuracy of reporting. This was achieved through a stronger and more accurate error management module, which involved thorough testing and the redefinition of some code components.
- Reporting modules were revamped and further debugged in order to better accommodate the new mobile aspects (final mobileOK assessments and new delivery contexts) and its articulation with the reporting of the WCAG 2.0 evaluation.
- All the bugs found in the pilot tests and further unitary tests were corrected
- Integration with the WCAG2 modules and GUI were improved. This involved some major modifications on the Web Services particularly deriving from the abovementioned revamping process.

6. Description Language Assessment module

6.1. Short description of Description Languages Assessment Module

The Description Languages Assessment Module consists of the following three sub-modules:

- The **Accessibility Threshold Controller** that is responsible for connecting to the ACCESSIBLE ontologies and receive the relevant guidelines and values for the assessment process. It also obtains the thresholds that will be used for the evaluation process. Some of the defined accessibility features include the size of GUI controls, the existence of text labels and titles, as well as their font size, the position of GUI components, etc.
- The **Accessibility Features Parser**, which is capable of parsing the SDL file and storing the accessibility semantic features. The module first communicates with the Accessibility Threshold Controller entity to get the application features that must be checked for the accessibility evaluation. It then parses the SDL system application's source code (files with extensions “.pkg”, “.pr” and “.fsm”), looks for specific SDL commands and seeks for the values of command parameters.
- The **Accessibility Evaluator Resultant** that is responsible for comparing the pre-defined thresholds with the current values of the accessibility features of the SDL system, displaying and storing the results of the accessibility evaluation and making the suggestions to increase application's accessibility. It provides input for the generation of the accessibility assessment Report (in printable as well as EARL based format). It also makes suggestions on how to increase the accessibility level of the SDL application. To satisfy the above functionality, the module communicates with the Accessibility Features Parser and Accessibility Threshold Controller modules. The following Figure depicts an overview of the description languages accessibility assessment module architecture.

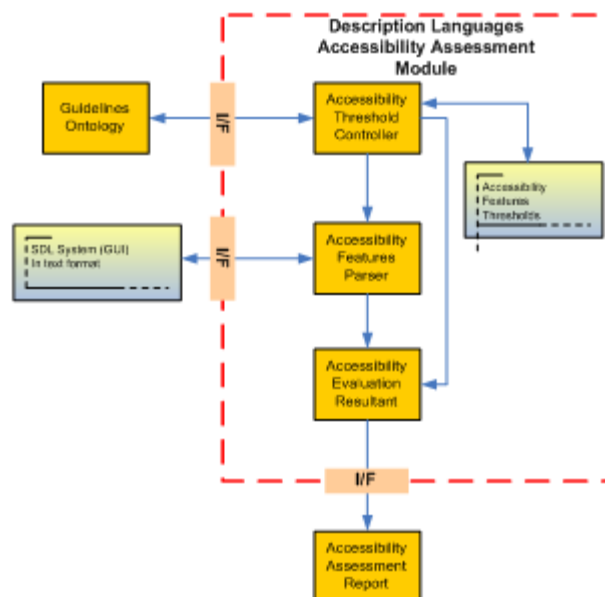


Figure 85: System Architecture of the Accessibility Assessment Module for SDL Systems.

6.2. Technologies related to Description Languages Assessment Module

6.2.1. Specification and Description Language (SDL) and relevant technologies

As already mentioned, the prototype of the Description Languages Assessment Module allows the accessibility assessment of SDL (Specification and Description Language) applications.

SDL is a specification language targeted at the unambiguous specification and description of the behaviour of reactive and distributed systems. It is defined by the ITU-T (Recommendation Z.100.). Although the ITU Specification and Description Language is widely used in the telecommunications field, it is also now being applied to a diverse number of other areas ranging over aircraft, train control, medical and packaging systems. It is a general purpose description language for communicating systems. The basis for description of behaviour is communicating Extended State Machines that are represented by processes. Communication is represented by signals and can take place between processes or between processes and the environment of the system model. Some aspects of communication between processes are closely related to the description of system structure.

For systems engineering the ITU specification and description language is usually used in combination with other languages: MSC, ASN.1, TTCN and UML. The use of the object model notation of SDL-2000 in combination with MSC, traditional Z.100 state models and ASN.1 is a powerful combination that covers most aspects of system engineering. This set of notations meets criteria for UML. There has been work relating the ITU specification and description language and TTCN semantic models. TTCN is often used for testing or validating standards or systems written using the ITU specification and description language. Similarly the ITU specification and description language is often used for systems or standards to be tested or validated using TTCN. The ITU Z.105 and Z.107 standards define the use of SDL with ASN.1. The Z.109 standard defines a UML profile for SDL, and the Z.120 standard defines Message Sequence Charts. There is a supplement to Z.100 that gives a methodology which includes TTCN.

SAFIRE IDE Integration

The Description Languages Assessment Module has been integrated in the applications environment of SOLINET's SAFIRE Integrated Development Environment (IDE). SAFIRE Professional IDE is a fully integrated development & run-time environment optimized for the implementation, validation & observation of complex systems. It is used for a wide range of applications, such as SW components of various platforms, gateways, signalling testers & protocol analyzers. With over a decade of experience in the industry, the SAFIRE team has created a powerful, innovative tool chain based on international standards, such as SDL, MSC, ASN.1 &

TTCN (ITU-T, ETSI, ANSI, ISO). SAFIRE Professional has a graphical development environment for quickly creating, editing & building systems, test harnesses & test suites.

With SAFIRE's advanced testing features, systems can be validated to various levels of confidence, from top-level tests to detailed conformance tests according to international standards or pre-defined requirements. SAFIRE tests are automated, deterministic, reproducible & documented. There are SAFIRE compatible libraries, test suites, drivers & hardware available for a wide range of systems, for example mobile, internet, aerospace & trunk networks libraries for signalling systems.

The SAFIRE Professional tool chain (Figure 86) has a modular architecture as shown in the following diagram:

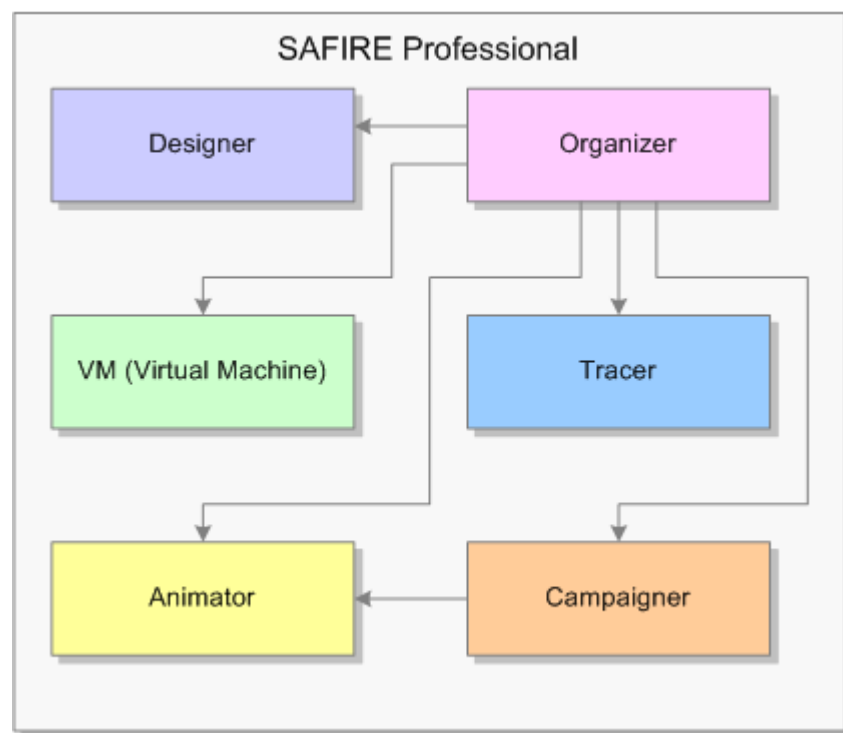


Figure 86: SAFIRE Tool Chain

The tool chain is composed of the following elements:

- SAFIRE Designer - graphical editor, viewer, compiler
- SAFIRE Campaigner - test execution & report generator
- SAFIRE Animator - slow motion replay (actions, events, behaviour)
- SAFIRE Tracer - Captures messages from the system's FSMs (finite state machines)
- SAFIRE Organizer - version control & project management
- SAFIRE VM Virtual Machine - high performance virtual machine

The entire SAFIRE tool chain is fully integrated using the same concepts, graphical representation, notation & data for the implementation, validation and observation of event-driven state machines.

SAFIRE Designer

The SAFIRE Designer (Figure 87) includes the following functionalities useful to efficiently design SDL systems:

- Graphical design of state machines & test suites
- The system architecture shows the state machine instances, their nesting, the connections between them and their interfaces.
- The behaviour of individual state machines is presented clearly as a flow chart including input events, outputs, actions, decisions, timers & resulting state transitions.
- The state-transition diagrams show the different states & all possible paths between the states, i.e. driven by input events.
- The state-input matrix presents all the states & the corresponding coverage of possible input events, i.e. highlighting which events are not handled.
- The arrow diagrams are an alternative view of the behavior showing input events, timers & outputs as arrows between the different FSMs.

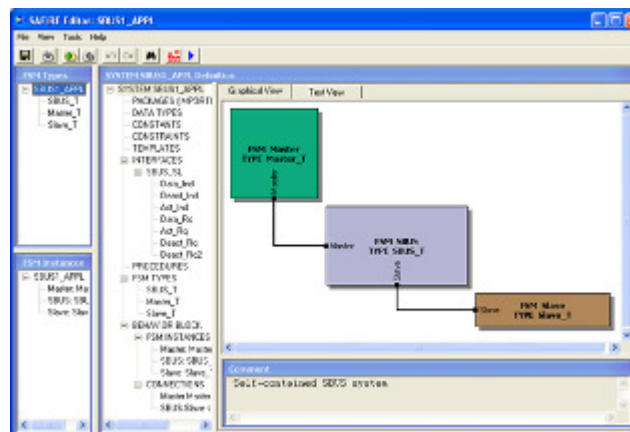


Figure 87: SAFIRE Designer

Validation: SAFIRE Campaigner – Animator- Tracer

SAFIRE validates software implementations using test harnesses running test scenarios, which are implemented using the same flow chart graphical representation as the SW components themselves (Figure 88).

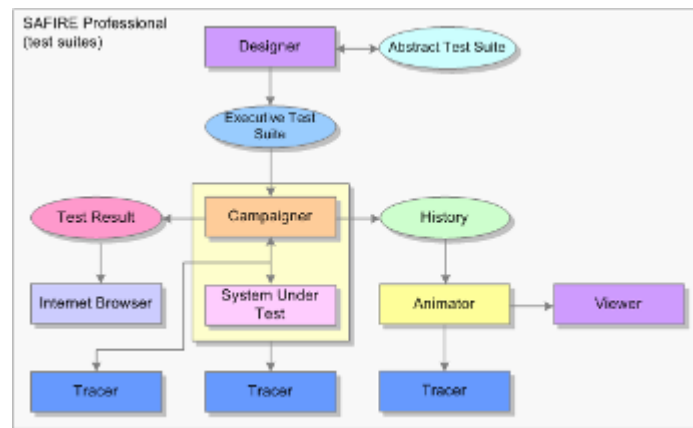


Figure 88: SAFIRE validation diagram

Related test scenarios can be defined and managed as test suites, and the SAFIRE Campaigner (Figure 89) allows test scenarios to be selected for test campaigns and automatically executed. The Campaigner runs test campaigns without human involvement and SAFIRE based tests meet the ISO requirements for conformance tests; tests are automated, deterministic, reproducible & documented.

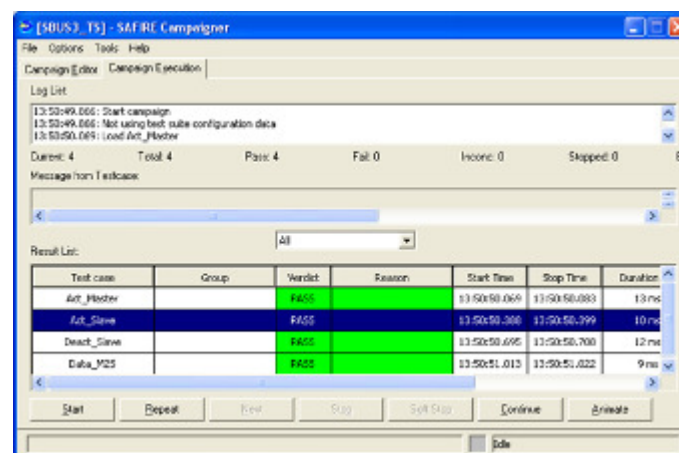


Figure 89: SAFIRE Campaigner

Information specific to individual test objects, such as appliance characteristics & services, can be configured without editing the test suites. Test results are updated during execution and can be saved for later analysis or exported as an HTML report for Internet publication. The SAFIRE Animator (Figure 90) can be launched from the Campaigner for a slow-motion, step-by-step, replay of any test executed. The Animator jumps directly to the points where test verdicts are set, allowing rapid analysis of the events leading up to a pass/fail result.

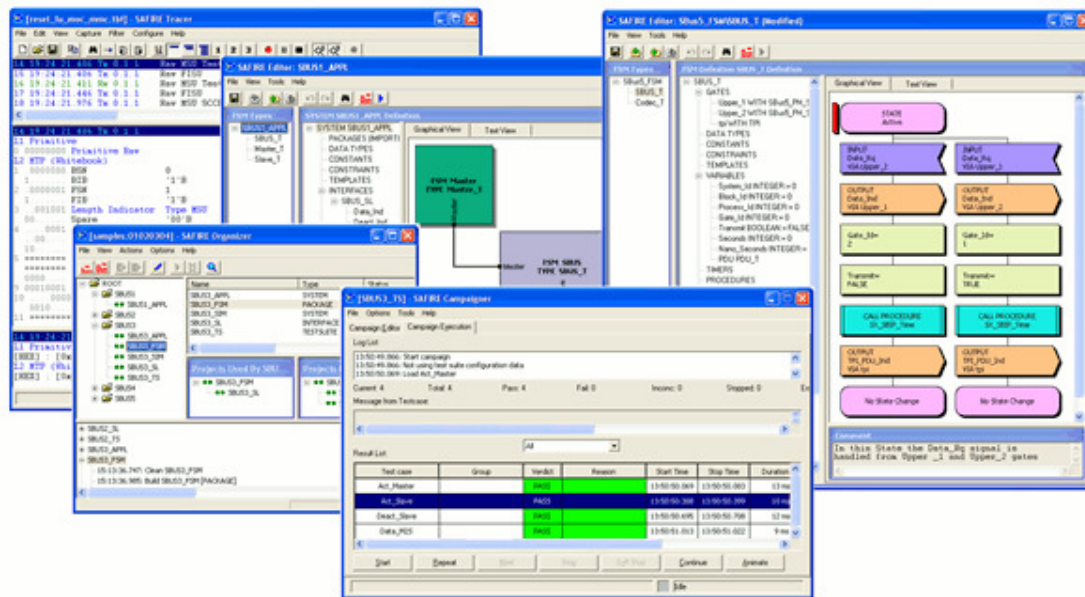


Figure 90: SAFIRE Animator

6.3. Technologies related to the implementation of the Description Languages Assessment Module

The development of the Description Languages Assessment Module tool was performed using the Java language (Java SE version 6.0) and the NetBeans Integrated Development Environment. A Web Service has been implemented for the communication of the module with the ACCESSIBLE portal using the Apache Axis framework.

The figure below is the Class Diagram of the Description Languages Assessment module. The system is constituted of four classes:

- FeaturesParser.java
- EvaluationResultant.java
- ThresholdController.java
- Result.java
- DescriptionLanguageWS.java – Web Service

The classes interact in an efficient way to provide the functionality of the system.

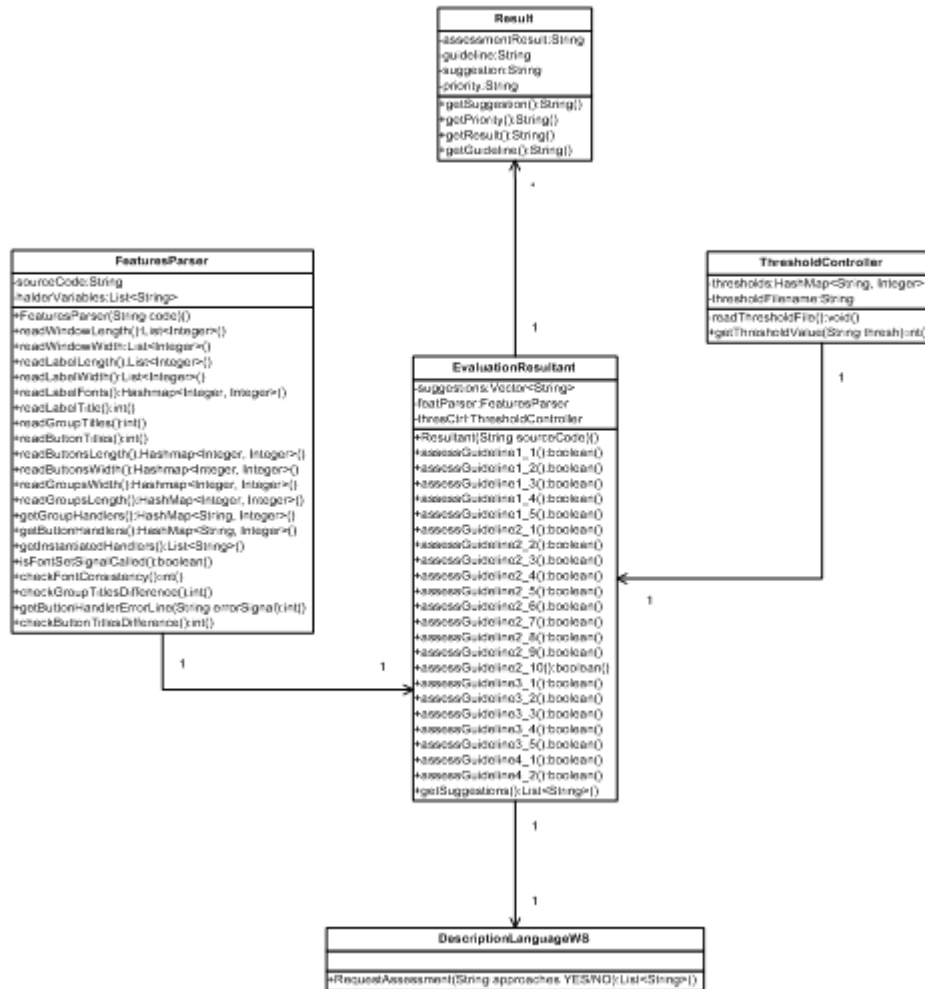


Figure 91: Class Diagram of the Description Languages Assessment Module

FeaturesParser.java

The FeaturesParser.java class provides all the required functionality for parsing the SDL source code. The class parses the .pkg files (SDL source code) and looks for certain signal calls. It then extracts the parameters' values of these signals and stores them in corresponding List objects. The EvaluationResultant class instantiates a FeaturesParser object to parse the SDL source code and get the values needed for the Assessment.

EvaluationResultant.java

The EvaluationResultant.java is responsible for performing the Assessments that the users of the System ask for. The class instantiates a ThresholdController and a FeaturesParser objects and uses them to obtain all the required for an Assessment data. It then performs the corresponding, for the accomplishment of the Assessments, comparisons.

ThresholdController.java

The class `ThresholdController.java` is responsible for reading and parsing the `DescriptionLanguage.thresholds` file in order to store the values of the Accessibility thresholds. A `ThresholdController` object is instantiated in the `EvaluationResultant` class to read and return the values contained in the file. The `EvaluationResultant` uses these values to perform the assessments.

Results.java

The class `Results.java` I used to create `Result` instances and store in an efficient way all the data of the assessment of a specific Guideline (i.e. guideline id, priority, result, suggestions). Then the instantiated `Result` type objects are added in a List that keeps together all the Assessment results.

DescriptionLanguageWS.java

The `DescriptionLanguageWS.java` is the class that creates the Web Service that allows the integration with the ACCESSIBLE portal. It provides the `RequestAssessment()` Web Method which is responsible of performing the Assessment. In fact the Web Service instantiates an `EvaluationResultant` object and sets it responsible for the accomplishment of the Accessibility Assessment.

6.4. Usage Scenarios for Description Languages Assessment Module (Standalone)

Selecting a development project and starting the Description Languages Assessment Module

The Description Languages Assessment Module may be launched from the SAFIRE IDE. After starting SAFIRE, the “Product Browser” screen is displayed (Figure 92).

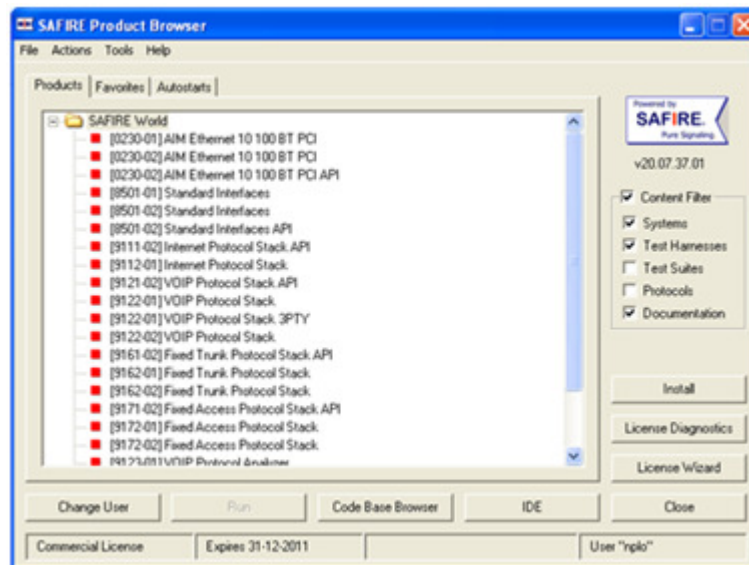


Figure 92: SAFIRE “Product Browser” screen.

SAFIRE IDE is launched by pressing the “IDE” button. The user is then prompted to select the archive (i.e. the development project) to be opened (Figure 93, Figure 94).

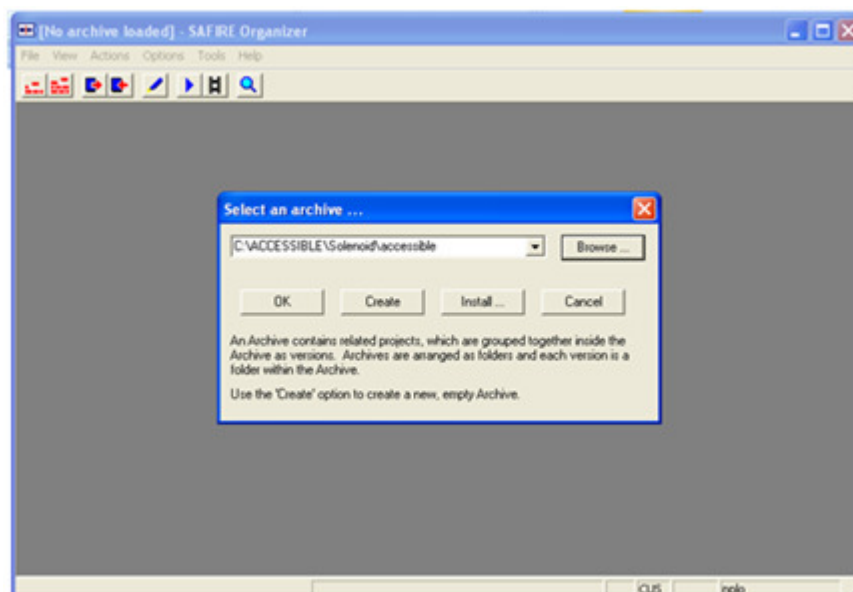


Figure 93: Selection of archive to be opened.

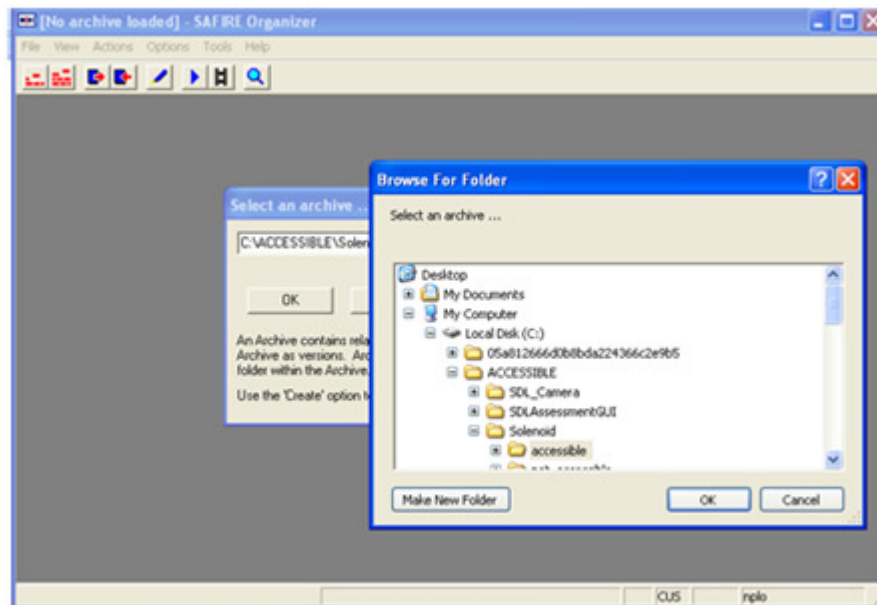


Figure 94: Browse for folder of the archive to be opened.

Following the selection of the archive, SAFIRE Organiser starts and the modules of the development project are displayed in the tree structure of the Organiser (Figure 95).

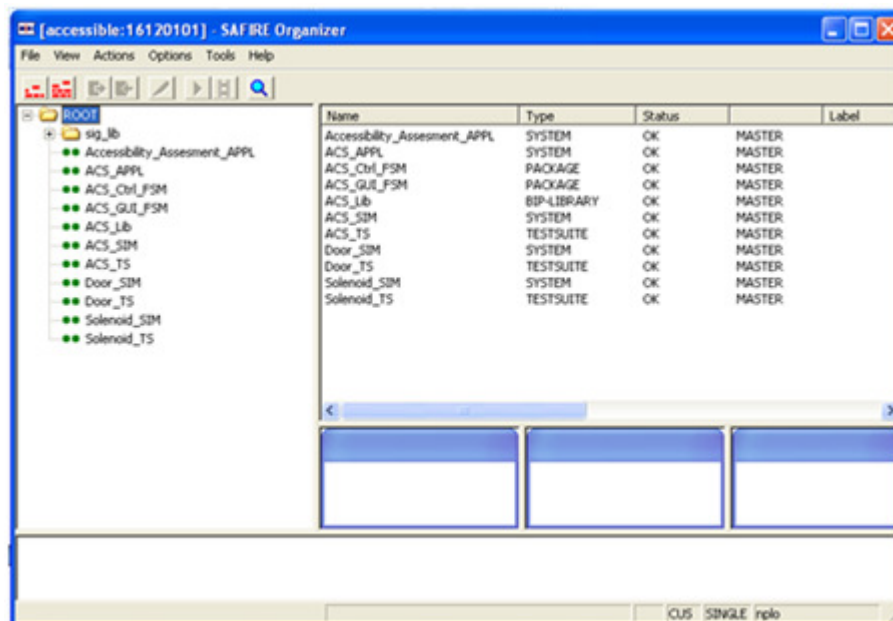


Figure 95: SAFIRE Organiser archive tree structure.

For each archive an “Accessibility_Assessment_APPL” application exists in the archive tree. The Description Languages Assessment Module for the archive is launched by right clicking on “Accessibility_Assessment_APPL” and selecting “Run” from the drop down menu (Figure 96).

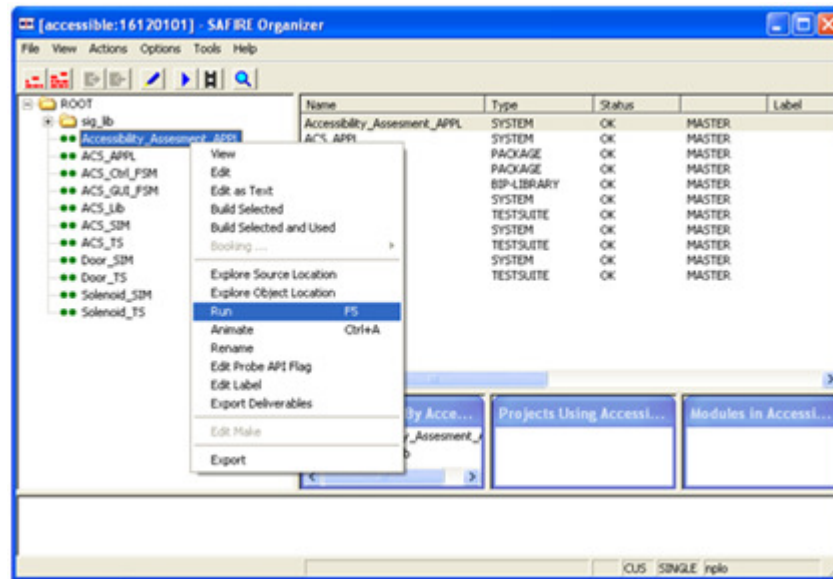


Figure 96: Launching the Description Languages Accessibility Assessment Module.

Selection of a preferable SDL application

The SDL GUI application of the specific archive is automatically pre-selected after launching Description Languages Assessment Module for the archive. As shown in Figure 97, the user can insert the local path of a different SDL source code file.



Figure 97: Selection of SDL application to be evaluated

Selection of the evaluation methodology

The next step involves the selection of the appropriate evaluation methodology that the user wants to perform for the evaluation assessment. The first method supports the manually selection of the supported Guidelines and respective evaluation Approaches. Thus, for each Guideline a number of approaches are suggested and they are grouped

according to the Level of Priority (Figure 98). It should be also noted that, as shown in Figure 98, Guidelines are grouped under Principles.

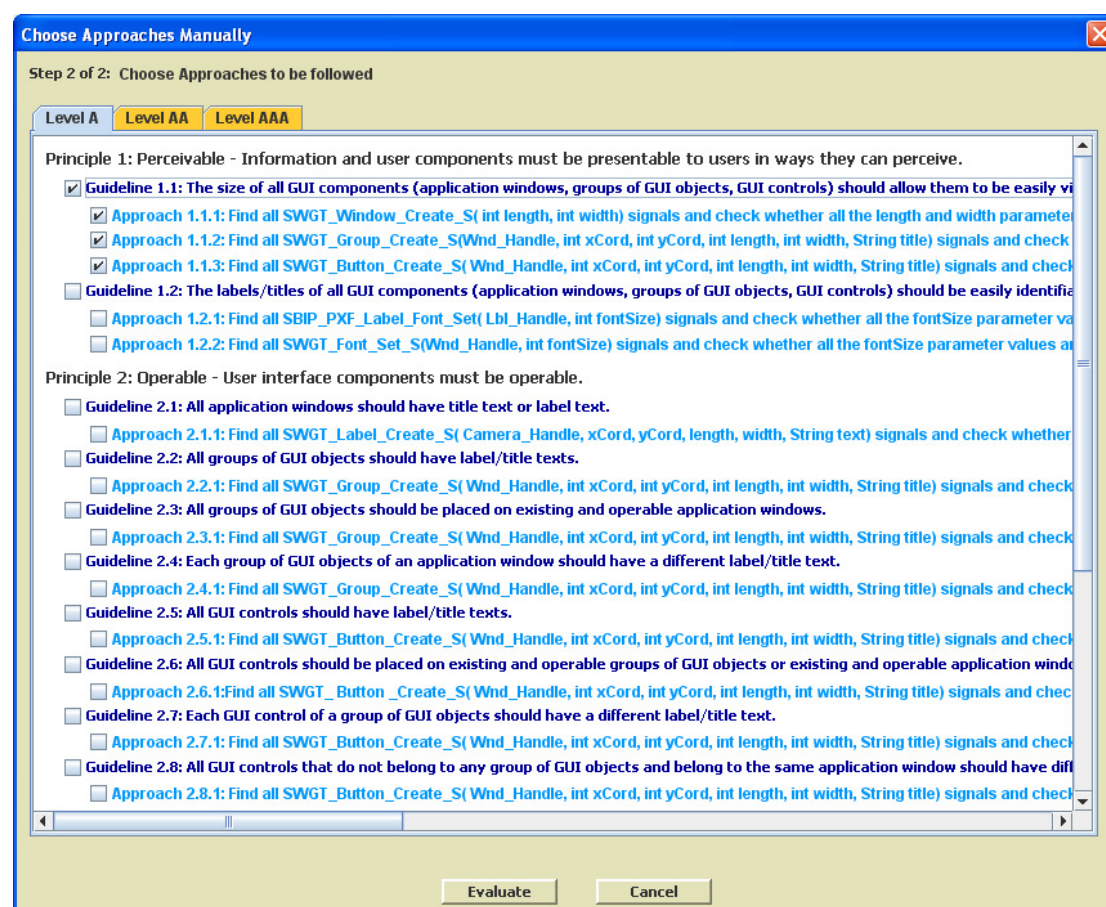


Figure 98: Manual selection of approaches

After the procedure of selecting the preferable approaches, the user initiates the process of the evaluation by pressing the button “Evaluate”.

Evaluation Results of the SDL assessment module

When the process is completed, the evaluation results are presented to the user (Figure 99). The results indicate the assessment result for each Guideline (*Results* information box in Figure 99). In addition, the tool provides detailed information on the detected accessibility limitations and provides suggestions for corrections indicating the line number where the related accessibility limitations have occurred.

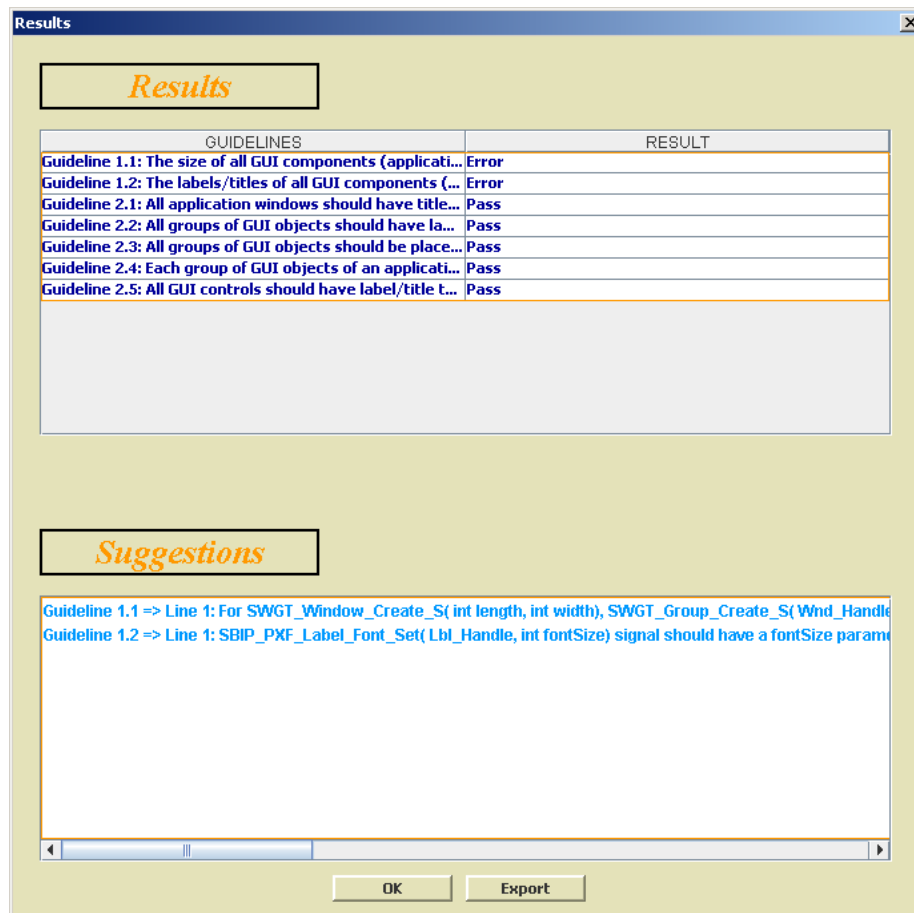


Figure 99: Results

The evaluation results can be exported by the EARL tool as depicted in the following figure.

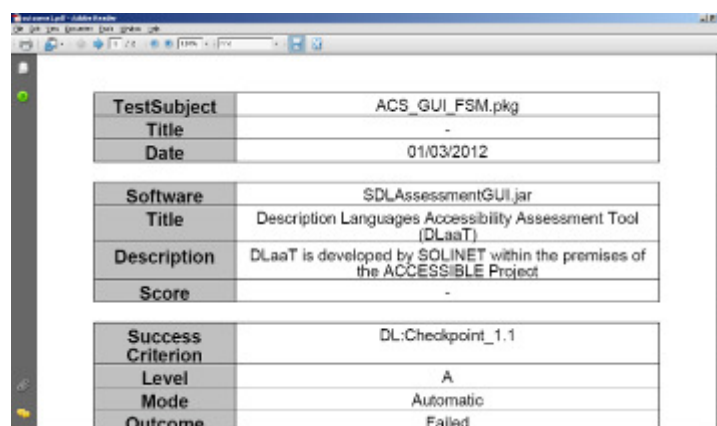


Figure 100: Results using the EARL tool

HAM-based Evaluation

As an additional evaluation procedure the ACCESSIBLE harmonized methodology (HAM) is being supported by the assessment module (Figure 101).



Figure 101: Selection of approaches based on the ACCESSIBLE harmonized methodology

Based on the *HAM methodology*, the following steps are necessary for specifying which approaches are needed for the evaluation process, after pressing the button “Choose approaches using ACCESSIBLE harmonized methodology” (Figure 101):

1. The user selects the preferable impairment (*Impairments* Tab in Figure 102)
2. For the selected impairment the user selects one or more associated disabilities (*Disability* Tab in Figure 103)
3. After the preferable impairment and disabilities are selected, the button “Find Approaches” is pressed and the tool responds with the corresponding approaches that apply (Figure 104)

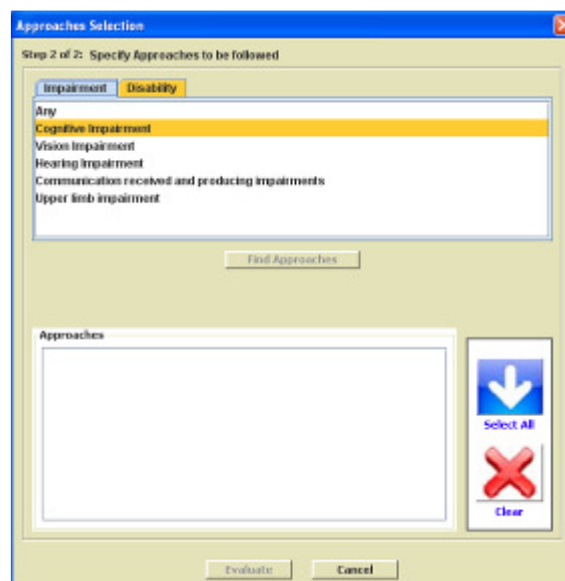


Figure 102: Selection of impairment

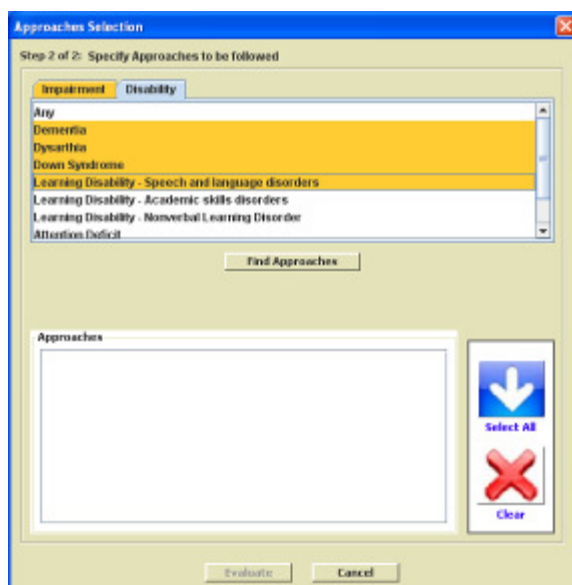


Figure 103: Selection of disabilities

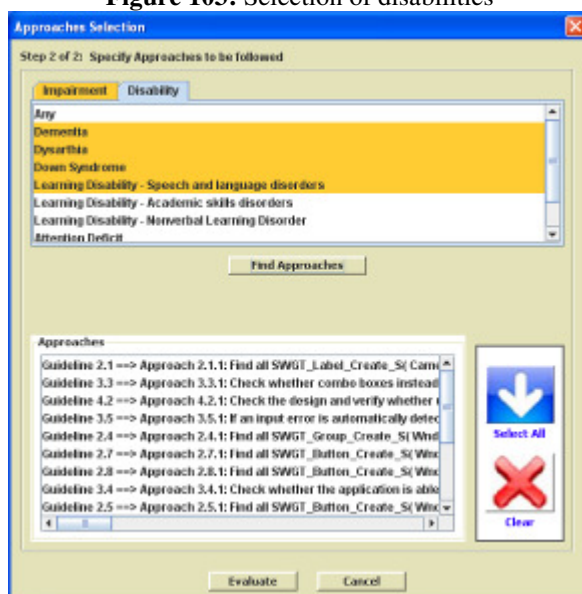


Figure 104: Listing of Guidelines/Approaches applying for the selected impairment / disabilities.

The user can then select the preferable approaches.

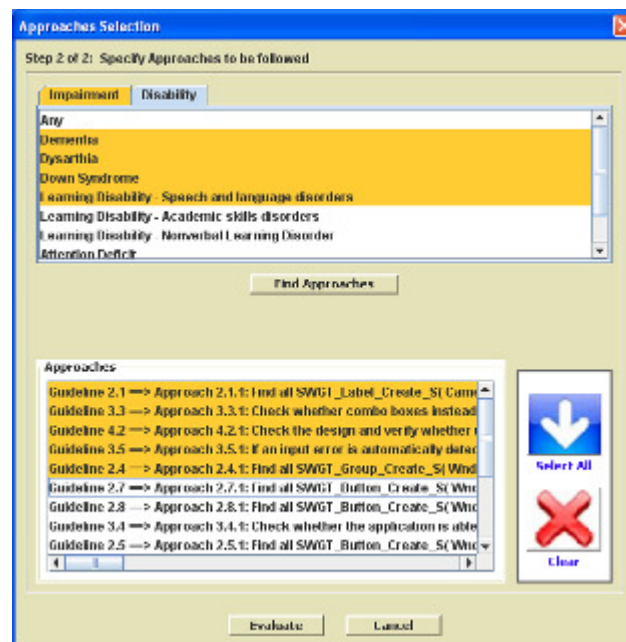


Figure 105: Selection of Approaches.

When the “Evaluate” button is pressed, the evaluation process starts and the evaluation results are presented to the user.

6.5. Usage Scenarios for Description Languages Assessment Module (ACCESSIBLE Portal)

Selecting accessibility assessment options for a specific project

After selecting a description languages assessment project, the user is prompted to select the accessibility assessment method to be used (Figure 106). The following options are available:

1. Evaluate a set of SDL files against personas
2. Evaluate a set of SDL files against specific guidelines
3. Evaluate a set of SDL files against ontology
4. Quick evaluation against ontology



Figure 106: Selection of the accessibility assessment method to be used.

Accessibility evaluation against personas

The following steps are performed after the user has selected option 1 “Evaluate a set of SDL files against personas” for a description languages assessment project. The user is presented with the profiles of existing personas and prompted to select personas for accessibility assessment.



Figure 107: Selection of personas.

After selecting personas and pressing the “Next” button, the user is presented with the available set of Evaluation Approaches and prompted to select the ones to be used for accessibility assessment (Figure 108). The Evaluation Approaches corresponding to the profiles of the selected personas have been automatically preselected.

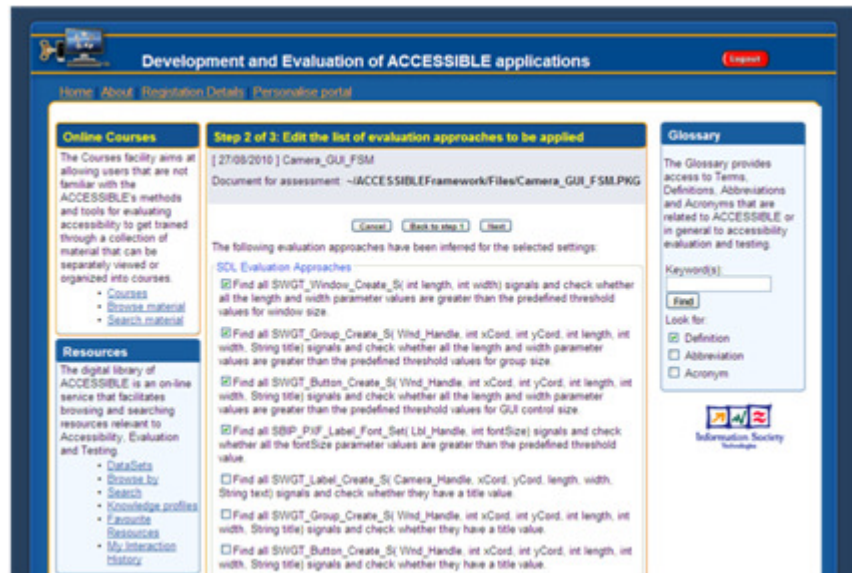


Figure 108: Selection of Evaluation Approaches.

After selecting Evaluation Approaches and pressing the “Next” button, the user is presented with the Accessibility Assessment Results for the concerned SDL application files (Figure 109). The accessibility assessment result in relation to each associated Guideline is presented. If the result is “Error”, then a suggestion for correcting the identified limitations is provided.

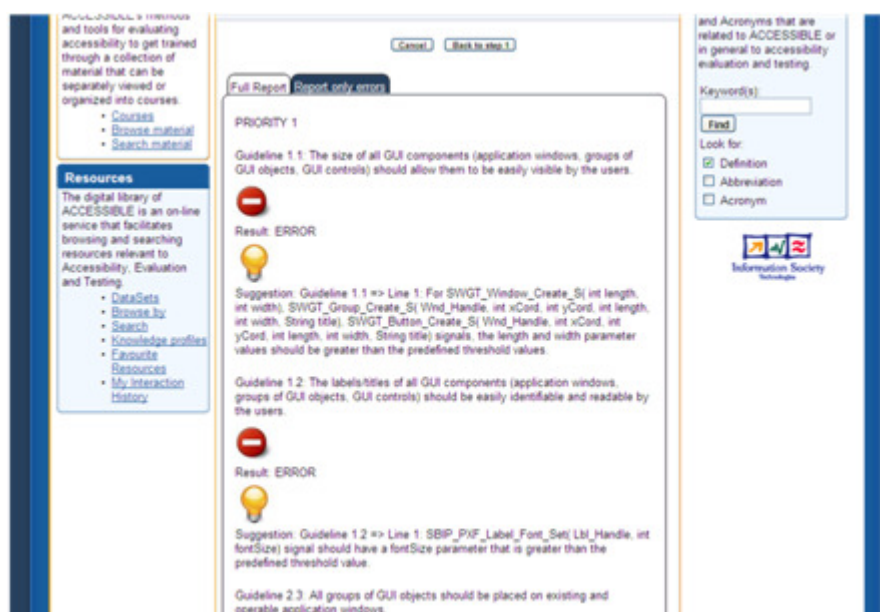


Figure 109: Accessibility Assessment Results.

Accessibility evaluation against specific guidelines

The following steps are performed after the user has selected option 2 “Evaluate a set of SDL files against specific guidelines” for a description languages assessment project. The user is presented with the set of defined Guidelines and associated Checkpoints for each Guideline, and prompted to select the Checkpoints for accessibility assessment.

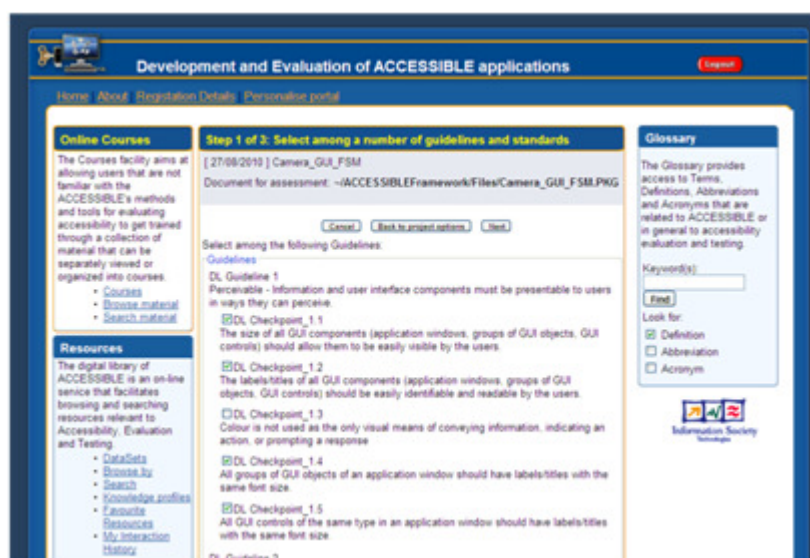


Figure 110: Selection of Checkpoints for each defined Guideline.

After selecting Checkpoints and pressing the “Next” button, the user is presented with the available set of Evaluation Approaches and prompted to select the ones to be used for accessibility assessment (Figure 110). The Evaluation Approaches corresponding to the selected Checkpoints have been automatically preselected.

After selecting Evaluation Approaches and pressing the “Next” button, the user is presented with the Accessibility Assessment Results for the concerned SDL application files. The accessibility assessment result in relation to each associated Guideline is presented. If the result is “Error”, then a suggestion for correcting the identified limitations is provided.

Accessibility evaluation against ontology

The following steps are performed after the user has selected option 3 “Evaluate a set of SDL files against ontology” for a description languages assessment project. The user is prompted to select the Disabilities (Figure 111), Functional Limitations (Figure 112) or Impairments (Figure 113) against which accessibility assessment will be carried out.

The screenshot shows a web application titled "Development and Evaluation of ACCESSIBLE applications". The main content area is titled "Step 1 of 3: Select among a number of evaluation categories for the ontology". Below this, it says "[27/06/2010] Camera_GUI_FSM" and "Document for assessment: --IACCESSIBLEFrameworkFiles/Camera_GUI_FSM.PKG". There are three buttons: "Cancel", "Back to assessment options", and "Next". Below these buttons are five tabs: "Disability", "Functional Limitation", "Impairment", "ATs", and "Software". The "Disability" tab is selected. It contains a list of checkboxes for various disabilities: Night blindness (checked), Deafness, Cerebral palsy, Deaf-Blindness, Arthritis, Asperger's syndrome, Quadriplegia, Dyspraxia (checked), Speech and language disorders (checked), Multiple sclerosis, Academic skills disorders, Blindness, and Deafness (checked). At the bottom, there is a small text "DTPubmedia Unit translation".

Online Courses
The Courses facility aims at allowing users that are not familiar with the ACCESSIBLE's methods and tools for evaluating accessibility to get trained through a collection of material that can be separately viewed or organized into courses.
• Courses
• Browse material
• Search material

Resources
The digital library of ACCESSIBLE is an on-line service that facilitates browsing and searching resources relevant to Accessibility, Evaluation and Testing.
• DataSets
• Browse by
• Search
• Knowledge profiles
• Favorite Resources
• My Interaction history

Glossary
The Glossary provides access to Terms, Definitions, Abbreviations and Acronyms that are related to ACCESSIBLE or in general to accessibility evaluation and testing.
Keyword(s):
Find
Look for:
☒ Definition
☐ Abbreviation
☐ Acronym
Information Society Technologies

Figure 111: Selection of Disabilities.

The screenshot shows the same web application as Figure 111, but now the "Functional Limitation" tab is selected. It contains a list of checkboxes for various functional limitations: Binocular acuity of distant vision (checked), Stability of joint functions, Colour vision, Expression of language, unspecified, Mental functions of language, other specified, Mobility of joint functions, unspecified (checked), Integrative language functions, Speed of speech (checked), Reception of language (checked), Voice functions, other specified, Thought functions, unspecified, Control of complex voluntary movements, and Insight (checked).

Online Courses
The Courses facility aims at allowing users that are not familiar with the ACCESSIBLE's methods and tools for evaluating accessibility to get trained through a collection of material that can be separately viewed or organized into courses.
• Courses
• Browse material
• Search material

Resources
The digital library of ACCESSIBLE is an on-line service that facilitates browsing and searching resources relevant to Accessibility, Evaluation and Testing.
• DataSets
• Browse by
• Search
• Knowledge profiles
• Favorite Resources
• My Interaction history

Glossary
The Glossary provides access to Terms, Definitions, Abbreviations and Acronyms that are related to ACCESSIBLE or in general to accessibility evaluation and testing.
Keyword(s):
Find
Look for:
☒ Definition
☐ Abbreviation
☐ Acronym
Information Society Technologies

Figure 112: Selection of Functional Limitations.

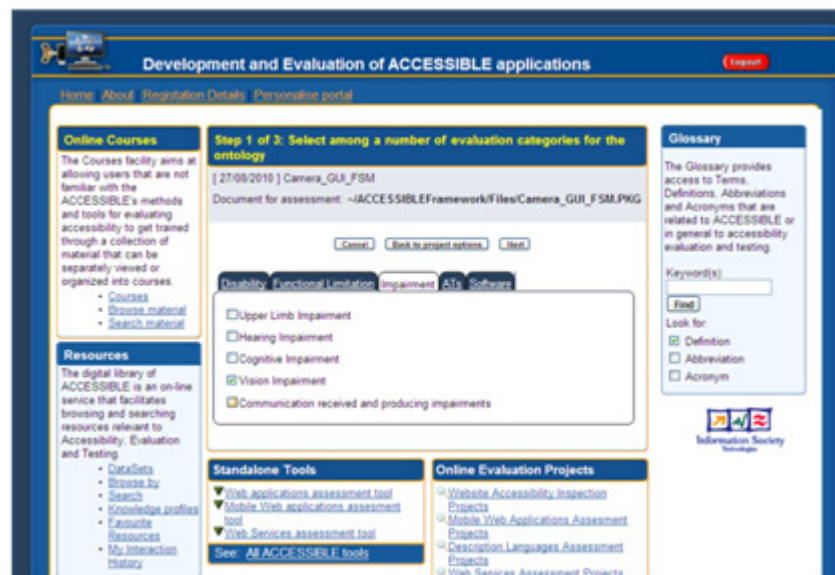


Figure 113: Selection of Impairments.

After selecting evaluation categories as illustrated above, the user is presented with the available set of Evaluation Approaches and prompted to select the ones to be used for accessibility assessment. The Evaluation Approaches corresponding to the selected evaluation categories have been automatically preselected. After selecting Evaluation Approaches and pressing the “Next” button, the user is presented with the Accessibility Assessment Results for the concerned SDL application files. The accessibility assessment result in relation to each associated Guideline is presented. If the result is “Error”, then a suggestion for correcting the identified limitations is provided.

Quick accessibility evaluation against ontology

The following steps are performed after the user has selected option 4 “Quick evaluation against ontology” for a description languages assessment project. The user is prompted to select the evaluation categories for accessibility assessment (Figure 114).

After selecting evaluation categories as illustrated above, the user is presented with the available set of Evaluation Approaches and prompted to select the ones to be used for accessibility assessment.

After selecting Evaluation Approaches and pressing the “Next” button, the user is presented with the Accessibility Assessment Results for the concerned SDL application files. The accessibility assessment result in relation to each associated Guideline is presented. If the result is “Error”, then a suggestion for correcting the identified limitations is provided.

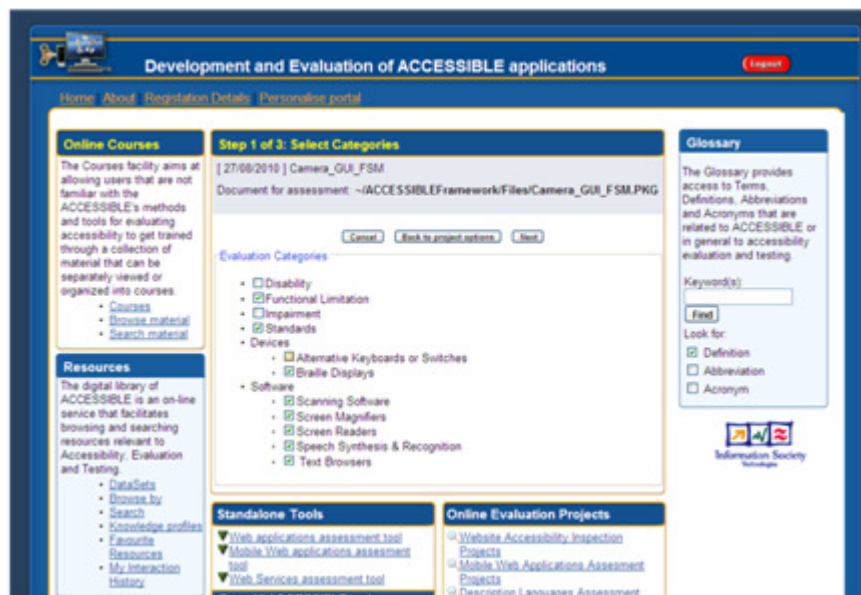


Figure 114: Selection of evaluation categories for quick evaluation.

6.6. Changes in current version

Since the last issued version of the tool a number of changes were made that are summarised in the following:

- The implementation of the DLaaT integration interface with the EARL-based reporting tool was modified so that the reported bug relating to the technique level was corrected. The “Level” field in the generated pdf is now correct for all techniques.
- The description of the Techniques was improved in order to more accurately describe the corresponding requirements and better be linked to the respective tips for correcting limitations.
- The description of the provided Tips was modified in order to more accurately point out the required modifications towards correcting accessibility limitations.
- An “accessible” and a “not accessible” version of the Solenoid SDL application were created towards providing an example of correcting accessibility limitations through the use of the tool.
- It was found that when a single disability was selected, then the mapping was correct. However, in the cases that multiple disabilities were selected, then, indeed, the mapping was wrong. The implementation was corrected and the mapping is now according to the HAM specification in the cases of multiple disabilities selection.

7. Introduction to the Disability Impairment Approximation Simulator (DIAS)

The ACCESSIBLE developer/designer-aid module, namely Disability Impairment Approximation Simulator (DIAS), is the one responsible for supporting target users (developers and/or designers) to facilitate the design and development of accessible Java Swing software applications. It approximately simulates the difficulties someone with specific vision, motor, cognitive and hearing impairments face, when interacting with Java Swing Graphical User Interfaces (GUIs) as well as Java ME applications. It communicates with the standalone application module or the NetBeans IDE depending on whether the user is using the standalone or the NetBeans plugin version of the developer/designer-aid module.

The purpose of ACCESSIBLE simulator is to assist developers and designers to better empathise with those who have reduced capabilities, and to help understand how capability loss affects the ability to interact with software applications and services. The module has been designed and developed for the following objectives:

- a developer-designer aid module for assistance to complete user centered design and accessibility simulation of how a Java Swing application can be viewed for users with impairments
- a Self-learning software, which can be obtained as a NetBean IDE plugin and/or as a standalone application, in order to present accessibility drawbacks and visual content problems of Java Swing applications

Thus DIAS is actually available in two versions, one standalone and one that is used as a NetBeans plugin. The version that can be used as a plugin on NetBeans, has two main functionalities. One that enables the developers/designers to *preview* their implemented GUI applications in a simulated fashion and also alerts them for any accessibility errors/warning the application may have. The developer/designer module is then able to provide appropriate recommendations for resolving any of these identified errors/warnings. On the other hand, the DIAS “*Run plugin*” gives the developers/designers the ability to execute and run in real time their Java Swing applications and verify if the supported functionalities and components contain any accessibility constraint.

The input to DIAS is a Java Swing GUI application bundled in a jar file or a Java Swing GUI form. Through the tool, the user can select the impairments that he/she prefers to simulate as well as specific corresponding controls for modifying the severity level for each of the selected impairment. Finally the output of this tool is the result of the simulation process with additional accessibility recommendations/tips. More information about DIAS, concerning system requirements, related technologies and the installation of DIAS plugin version in NetBeans can be found in the Deliverable D5.3 “Developer & designer’s aid Module”.

7.1. Impairments and symptoms covered by DIAS

Prior to building an accessibility simulation tool, it is very important to identify its target impairments, as well as their symptoms and parameters that need to be modelled. To this end, our work started from an appropriate analysis of relevant disabilities that needed to be covered by our tool, towards modelling their most common symptoms within a proper framework that would form the basis for the desired simulations.

Our research work focused over the simulation of impairments that are relevant to HCI, in terms of being capable to significantly hinder it. Modern HCI is heavily based on the user's visual perception of the application's user interface (UI) and also on the user's capability to provide input to the computer through a mouse or a keyboard. In some cases, the user should also perceive sounds produced from the application. It is thus clear that visual, hearing, physical and of course, cognitive deficiencies can pose significant obstacles in HCI.

Focusing over these categories of impairments, a largescale literature survey was conducted, towards identifying impairments that were relevant to our scope, both commonly addressed and also well-documented so as for appropriate filters simulating their most common symptoms to be developed.

The specific impairments that were finally selected to be addressed from our tool are briefly described in the following paragraphs:

Macular Degeneration is a condition that damages the macula, the central part of the retina. The macula is responsible for central vision and the ability to see details. When the macula is damaged, the eye loses its ability to see details, such as small print, facial features or small objects.

Glaucoma is the term for a diverse group of eye diseases, all of which involve progressive damage to the optic nerve. Optic nerve damage produces certain characteristic visual field defects in the individual's peripheral (side), as well as central, vision.

Cataract is the term generally used for an opacification or a discoloration of the lens substance. It may be localized in certain lens parts or may interfuse the whole lens. According to its localization, the opacity causes a more or less pronounced disturbance of light transmission.

Hyperopia, also known as farsightedness, longsightedness or hypermetropia, is a defect of vision caused by an imperfection in the eye (often when the eyeball is too short or the lens cannot become round enough), causing difficulty focusing on near objects, and in extreme cases causing a sufferer to be unable to focus on objects at any distance.

Colour Vision Deficiencies. *Protanomaly*: Having a mutated form of the long-wavelength (red) pigment, whose peak sensitivity is at a shorter wavelength than in the normal retina, protanomalous individuals are less sensitive to red light than normal.

Deuteranomaly: This is the most common form of colour blindness. The deuteranomalous person is considered "green weak".

Tritanomaly: Regards a mutated form of the short wavelength (blue) pigment. The short-wavelength pigment is shifted towards the green area of the spectrum. This is the rarest form of anomalous trichromacy colour blindness.

Protanopia: Lacking the long – wavelength sensitive retinal cones, those with this condition are unable to distinguish between colours in the green yellow red section of

the spectrum. They have a neutral point at a greenish wavelength around 492nm that is, they cannot discriminate light of this wavelength from white.

Deuteranopia: Lacking the medium - wavelength cones, those affected are again unable to distinguish between colours in the green yellow red section of the spectrum. Their neutral point is at a slightly longer wave-length, 498nm.

Tritanopia: Lacking the short - wave-length cones, those affected are unable to distinguish colors along the blue yellow dimension.

Night Blindness (Nyctalopia) is a condition making it difficult or impossible to see in relatively low light. It is a symptom of several eye diseases. It can be described as insufficient adaptation to darkness.

Extreme Light Sensitivity (Photophobia) is a symptom of abnormal intolerance to visual perception of light.

Retinitis pigmentosa (RP) is a group of genetic eye conditions that leads to incurable blindness. In the progression of symptoms for RP, night blindness generally precedes tunnel vision by years or even decades.

Parkinson's disease is a degenerative disorder of the central nervous system. Early in the course of the disease, the most obvious symptoms are movement - related; these include shaking, rigidity, slowness of movement and difficulty with walking and gait.

Dyslexia is a very broad term defining a learning disability that impairs a person's fluency or comprehension accuracy in being able to read, and which can manifest itself as a difficulty with phonological awareness, phonological decoding, orthographic coding, auditory short-term memory, or rapid naming.

Conductive hearing loss occurs when there is a problem conducting sound waves anywhere along the route through the outer ear, tympanic membrane (eardrum), or middle ear (ossicles). It may occur in conjunction with sensorineural hearing loss or alone. It is caused by any condition or disease that impedes the conveyance of sound in its mechanical form through the middle ear cavity to the inner ear. A conductive hearing loss can be the result of a blockage in the external ear canal or can be caused by any disorder that unfavourably effects the middle ear's ability to transmit the mechanical energy to the stapes footplate. This results in the reduction of one of the physical attributes of sound called intensity (loudness), so the energy reaching the inner ear is lower or less intense than that in the original stimulus. Thus, a reduction in sound level, or the ability to hear faint sounds is encountered.

Sensorineural hearing loss (SNHL) is a type of hearing loss, in which the root cause lies in the vestibulocochlear nerve, the inner ear, or central processing centers of the brain. It thus results from inner ear or auditory nerve dysfunction. The sensory component may result from damage to the organ of Corti, or an inability of the hair cells to stimulate the nerves of hearing, or a metabolic problem in the fluids of the inner ear. The neural or retrocochlear component can be the result of severe damage to the organ of Corti that causes the nerves of hearing to degenerate, or it can be stem from inability of the hearing nerves themselves to convey neurochemical information, through the central auditory pathways. Like conductive hearing loss, it reduces the intensity of sound, but it can also lead to distortion of the perceived sound.

The common symptoms of the above impairments, as reported in the relevant literature (several indicative of which are presented in Table 6), formed the basis towards the development of filters that would simulate their impact during HCI. The development of these filters consisted of two phases. The first phase involved the initial implementation of visual, motoric, hearing and cognitive filters, on the basis of parameters, such as the ones presented in Table 6. These initial filters were evaluated

from expert doctors and patients, who provided feedback over whether they were appropriate and whether modifications were needed over them, so as for more realistic simulations to be provided. On the basis of this feedback, the final version of the filters was implemented.

Disability	Parameters and abnormal values identified from literature
Macular Degeneration	- Visual acuity of 246 patients with age-related macular degeneration: a) 20/20-20/25 (60 patients), b) 20/30 to 20/40 (65 patients), c) 20/50 to 20/100 (57 patients) and d) less than or equal to 20/200 (65 patients) [22]. - Patients with MD generally fixate above or to left of their central scotoma [30]. - Fixation variability increases with scotomata size; for scotomata sizes greater than 20° average diameter, sharp increase in fixation variability can be noticed [52]. - Degree of eccentricity from the fovea: From 2° to 11° [29]. - Visual Field Defect (VFD) [49]
Glaucoma	- Diffuse depressions in the hill of vision. [34]. - For Chronic Angle Closure Glaucoma, nasal area damaged: 52% in the superior hemifield, 58% in the inferior hemifield. [38] - Paracentral scotomas in the central 30° of the visual field. [51] - Advanced glaucoma: large arcuate scotomata in the visual field [50]. - Reduced contrast sensitivity at the areas of visual field defects [40].
Cataract	- Glare sensitivity for people with cataract [25]: $GS = 1.2 * (C_g/C_0 - 1) * L/E$, where C_0=Contrast threshold, C_g =Contrast threshold with glare source, L=the mean screen luminance, E=the illuminance of the glare source at the eye. - For nuclear and cortical cataracts with a LogMAR visual acuity of less than 0.5 (Snellen equivalent better than 6/18), no loss of contrast sensitivity at the lowest spatial frequency (1c/deg). [25]
Hyperopia	- People with hyperopia can experience blurred vision, asthenopia, accommodative dysfunction, binocular dysfunction, amblyopia, and strabismus [14]
Protanomaly	- All three color pigments but deficiency in red responsiveness [41]. - Requires more red (671 m) and less green (535 m) to match the yellow than does a normal [44]. - Requires at least 1.5 times more red intensity than normal in the upper field to match a yellow in the lower field [41].
Deuteranomaly	- People with deuteranomaly have a mutated form of the medium-wavelength (green) pigment [36]
Tritanomaly	- A mutated form of the short-wavelength (blue) pigment [36]
Protanopia	- Difficulties to distinguish between blue/green and between red/green colors [27]. - Deuteranopes and protanopes do not accept the classical dichromatic matches when field size extends to 8° visual angle [47].
Deuteranopia	- Deuteranopic unilateral dichromats report that with only their deuteranopic eye open, they see wavelengths below the neutral point as blue and those above it as yellow [35]
Tritanopia	- The tritanope sees only two hues (red and green) in the spectrum [23] - Hybrids as orange and green-yellow or blue-green and purple are not included in a tritanopes spectrum [39]
Night blindness	- Spectral sensitivity data revealed presence of rod-mediated vision at wavelengths <540nm [45]. - Photopic abnormalities including abnormal visual acuity, a delayed final cone threshold or plateau, an elevated cone threshold, subnormal single-flash photopic responses, slow photopic implicit times and a low flicker fusion frequency [37].
Extreme light sensitivity	- Thresholds for photophobia were determined at wavelengths from 440 to 640 nm [48].
Retinitis pigmentosa	- Patients typically lose night vision in adolescence, side vision in young adulthood, and central vision in later life because of progressive loss of rod and cone photoreceptor cells [32]. - Photoreceptor function is diminished generally many years before symptomatic night blindness, visual-field scotomas, or decreased visual acuity arise [32]. - In [15], the mean change in visual function (visual acuity, visual field area, foveal ERG, Full-field ERG) over 3 years in study population with RP was presented.
Parkinsons disease	- The Parkinsonian tremor of DIAS was developed according to the mathematical model presented in [28].
Dyslexia	- Seeing letters or words backwards [46] - Often calling a b a d [46] - For readers who have dyslexia, extra time is an essential accommodation that allows them the time to decode each word [46]. - Regular symptoms of dyslexia include weakness in phonological processing, poor working memory, poor auditory discrimination, confusion over syntax and faulty auditory sequence [43].
Sensorineural hearing loss	- In [18], thresholds describing how much of the acoustical information in amplified speech is accessible to people with varying degrees of sensorineural hearing loss are presented.
Conductive hearing loss	- Conductive hearing loss, characterised by reduced efficiency of sound transmission through the external and/or middle ear generally produces a mild (26-44 dB HL) to moderate hearing loss [31].

Table 6: Indicative impairment parameters derived from the relevant literature, which were followed for the implementation of our simulation filters.

7.2. Implementation of simulations

Our developed tool is based on a series of filters, developed on the basis of the above theoretical background, which take as input the appearance of the simulated application and provide as output the results that would have been received by an impaired end user with the supported disabilities. As said, our developed filters enable the simulation of a series of visual, hearing, cognitive and motor impairments. Moreover, the joint simulation of visual and motoric impairments is supported from DIAS, as well as the simulation of impairments along with the use of assistive technologies, such as screen magnifiers and screen readers.

Visual impairments

Visual impairments are simulated within DIAS through filters based on the OpenCV library, which are applied in cascade over (continuously updated) images that depict the presentation of the UI under examination. In particular, given N symptoms that are involved in a given impairment, N filters are applied in cascade within each cycle of a filtering thread. At a modern PC, our filtering thread can produce in average over 20 cycles per second, thus providing a throughput of over 20 fps, adequate for realistic simulations.

Our tool is mainly developed in Java, thus, the JavaCV interface to the OpenCV libraries is utilized. This way, the advanced and fast image processing capabilities of OpenCV are ported in the Java environment, and are thereafter utilized, leading to the implementation of effective filters for the simulation of visual impairments.

Our developed vision filters can be regarded to basically belong in three categories: simulating (a) tunnel-vision symptoms and (b) their “inverse” (loss of central vision related symptoms), as well as (c) symptoms that regard the whole visual field. Moreover, the combination of these three categories of filters leads to the simulation of further disability symptoms.

It should be noted at this point that, for the simulation of tunnel-vision symptoms, like in the case of glaucoma, randomized semi-opaque masks (simulating scotomata) are applied on the vision field’s periphery (Figure 115.d where a VERITAS UI example was used), whereas as the symptoms’ severity level increase, the masks become less opaque and bigger. On the contrary, for the simulation of loss-of-central vision symptoms, similar masks are applied at the centre of the visual field (Figure 115.c). Similar masks can also be applied over the whole visual field (e.g. in the case of cataract, Figure 115.b).

Figure 115 presents how all supported visual impairments are simulated through DIAS, by taking as example an instance of a Java application, whose original, normal appearance is shown in Figure 115.a. In order to provide a clear view of the supported visual simulations, Figure 115 depicts simulations of impairment symptoms, regarded at an increased severity level. As shown in Figure 115, the simulation of cataract involves randomized semi-opaque masks that are applied in the whole visual field. Moreover, an openCV blur filter is applied so as to simulate the degradation of visual acuity. Also, manipulations are made over the hue and saturation of the UI’s appearance, in the HSV colour space, leading to the simulation of yellowing and saturation symptoms. Furthermore, the glare sensitivity symptom of cataract is also simulated through appropriate manipulations over the bright portions of the UI. Macular degeneration is simulated by DIAS through semi-opaque masks that are applied at the centre of the visual field. Moreover, the degradation of visual acuity symptom of this impairment is simulated

through image blurring filters. Glaucoma and retinitis pigmentosa are simulated through semi-opaque masks that are applied at the periphery of the visual field. Also, blurring filters are applied on the image depicting the examined UI. It should be noted however, that, since retinitis pigmentosa at severe levels affects the central visual field as well, in these cases, semi-opaque masks are also applied at the centre of the visual field, as shown in Figure 115.e.

Hyperopia (Figure 115.f) is simulated through blurring (Gaussian) filters. The simulation of Night blindness and extreme light sensitivity, involve filters that manipulate the examined whole UI presentation's brightness and contrast. Finally, in respect of colour blindness, the following impairments are supported: protanopia, deuteranopia, tritanopia, achromatopsia, protanomaly, deuteranomaly, tritanomaly and achromatomaly. These impairments are all simulated through filters, appropriately manipulating the colours of the image depicting the examined UI in the RGB and LMS colour spaces.

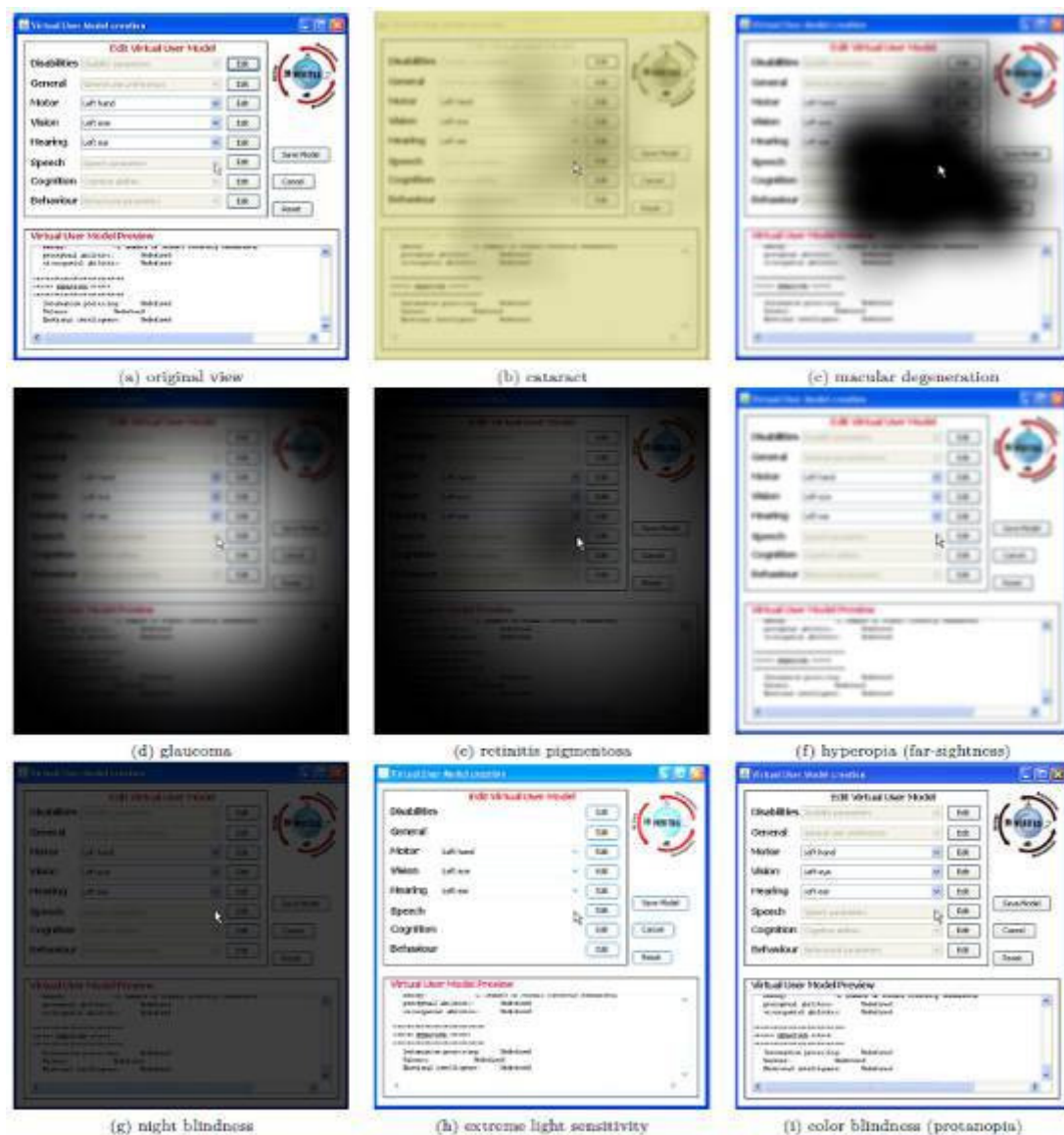


Figure 115: Simulation of visual impairments over the UI shown in Fig. (a)

Hearing impairments

Our developed tool supports also the simulation of hearing disabilities. This is done through audio filters, incorporated in DIAS, which provide the developer of an application with insight over how common hearing impairments affect the way an end user perceives sounds. Conductive hearing loss and sensorineural hearing loss impairments are supported from DIAS, through an audio filter that manipulates the loudness of the delivered sound.

Like conductive hearing loss, sensorineural hearing loss reduces the intensity of sound, but it might also introduce an element of distortion into what is heard resulting in sounds being unclear even when they are loud enough. Two types of sensorineural hearing loss are simulated within the DIAS tool, the “Noise Induced Hearing Loss” and the “Age Related Hearing Loss”, each regarded in mild and moderate-severe categories. In order to simulate these categories of sensorineural hearing loss, four different filters were implemented.

These filters cause a change to the frequency of the sound, resulting in distortions. Figure 116 depicts the filters that were implemented for each category of sensorineural hearing loss. The filters of Figure 116.a and Figure 116.b were used in the implementation of “Noise Induced Hearing Loss” for the mild and moderate-severe categories respectively. The ones of Figure 116.c and Figure 116.d were used in the implementation of “Age Related Hearing Loss”, in respect of the same categories. As said, a further filter leading to reduction in the volume of the sound is also applied within each category of sensorineural hearing loss.

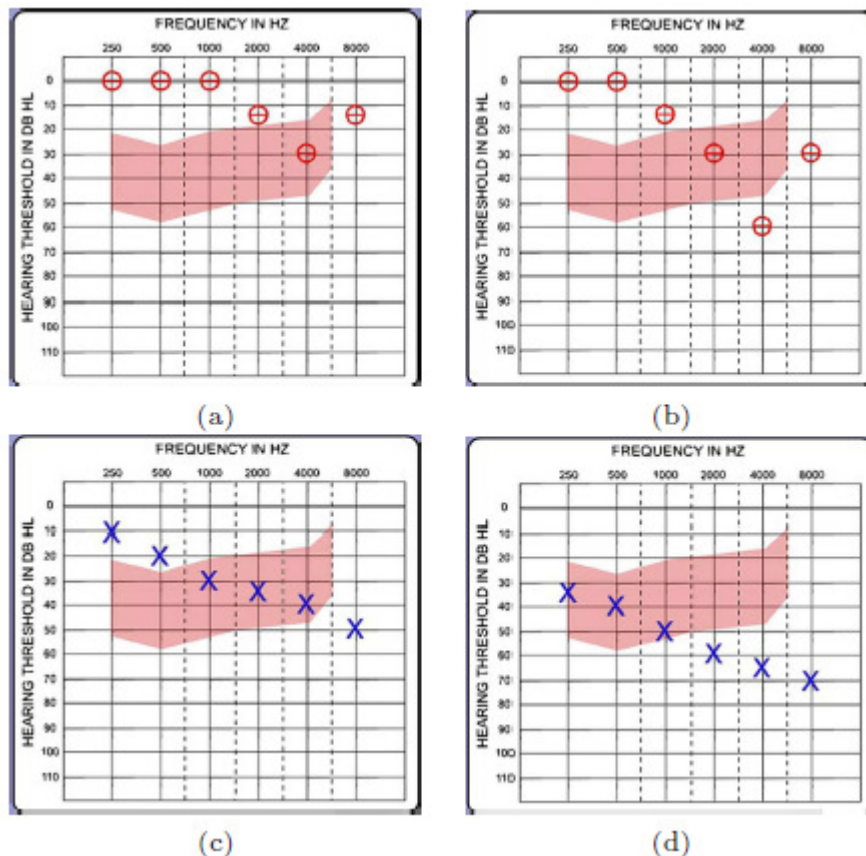


Figure 116: Filters implemented for various categories of sensorineural hearing loss: (a) mild noise induced hearing loss, (b) moderate-severe noise induced hearing loss, (c) mild age -related hearing loss, (d) moderate age -related hearing loss

Physical impairments

Physical impairments of the upper limbs (e.g. due to Parkinson's disease) can lead to problems during interaction with an end user interface, since in such cases, the end user may be for instance incapable to focus the mouse over small-sized elements (e.g. buttons) so as to click them. In order to provide developers with insight over the potential appearance of such issues within their developed UI, our developed tool embeds also a filter, capable to provide simulation over how the mouse of an end user with Parkinson would navigate. In order to do so, this specific filter takes as input the UI's appearance and the position of the mouse during the interaction. In the filter's output, the UI is presented along with the mouse pointer, whereas the latter is put in progressively updated pseudo-random positions, centred to the mouse pointer's actual position. For this purpose, a separate programmatic thread is utilized, which appropriately updates the position of the mouse on the basis of (a) the actual mouse position and (b) different frequency and amplitude deviations, in respect of the severity level of Parkinson simulated.

Moreover, DIAS provides also the capability to jointly simulate Parkinson-related symptoms and all visual/cognitive impairments that are supported from the tool. This is realized through the sequential application of visual/cognitive impairment filters and the one simulating parkinsonian tremor, over the examined application's UI.

As said, Parkinson can lead to deficiencies in the way an end user may focus over and select small-size UI elements. In this respect, our developed tool is also capable to identify such "problematic" elements, i.e. UI elements which should be resized, so as to become accessible to end users suffering from the currently simulated severity level

of Parkinson. As shown in Figure 117, these elements are marked in red within the output of the simulation. By clicking over these elements, the developer obtains further details over the error identified, and over how the specific element's size should be altered, so as to cater for the needs of end users with the present disability. In the example shown from Figure 117 a large number of elements (e.g. buttons) of the examined application are not large enough so as to allow an end user with severe Parkinson to easily click them.

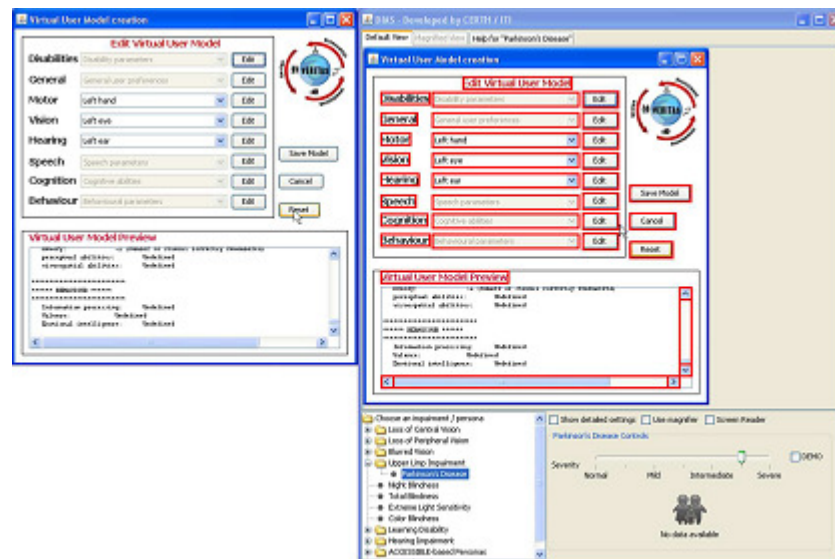


Figure 117: Instance of Parkinson simulation over a Java application

Cognitive impairments

Apart from visual, hearing and motoric impairments, our tool supports also the simulation of symptoms related to dyslexia. As shown in Figure 118, the dyslexia symptoms covered from our tool are: “confusing small words”, “letter reversals”, “word reversals”, “inversion of letters”, “mirroring of numbers” and “wrong order of letters”. These symptoms can be simulated either on their own, or in combinations, so as for the developer to examine how her/his development’s UI could be perceived from various dyslexic end users.

During dyslexia simulation, DIAS identifies UI elements containing text that is shown to the application’s user. It subsequently alters their text in respect of the simulated symptom(s). As a result, the filtered version of the UI contains new textual elements, with text that is accordingly altered (Figure 118). This way, the developer can easily identify potential misunderstandings that could be caused due to a dyslexic user’s difficulty to perceive textual info that is provided through the application’s UI.

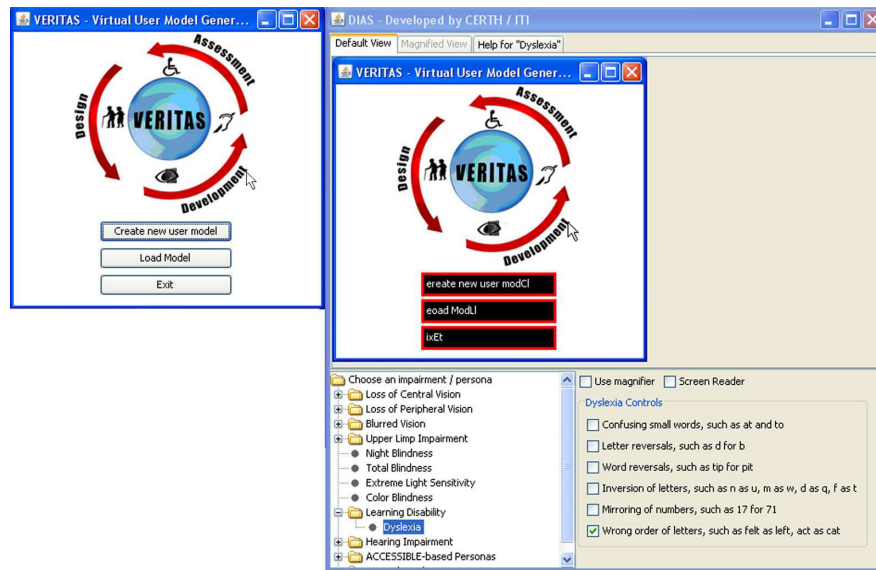


Figure 118: Instance of dyslexia simulation over a Java application

7.3. Personas

By incorporating the notion of personas (based on descriptions provided in the deliverables D2.3 and D2.4), our developed tool provides also practical examples of people that may suffer from the supported disabilities. Personas in the DIAS tool are virtual people, who suffer from specific disabilities. The tool is capable to provide simulations for a list of supported personas, with various visual, motoric, etc. impairments. Personal information about them, their disabilities symptoms, as well as the way these affect their daily life are all extensively described within DIAS. Thus, in case the developer has in mind specific users with disabilities that would be targeted from her/his UI, personas provide her/him with a more direct way of examining the UI against virtual people who suffer from symptoms relevant to the ones of the actual people in mind.

7.4. Incorporation of Magnifier and Screen reader within simulations

The afore described filters incorporated in DIAS offer a simulation framework, capable to provide UI developers an insight over how their developments' presentation may be affected from various disabilities. Nevertheless, modern HCI in respect of disabled users may as well be facilitated from assistive technologies, such as screen readers and screen magnifiers. For instance, a large variety of screen readers can be integrated in modern operating systems, such as JAWS or NVDA, providing visually impaired users with an audio description of the UI elements they are interacting with.

Moreover, screen magnification is an option typically incorporated in modern operating systems, allowing people with visual impairments to better understand the application in hand, thus significantly facilitating interaction. These facts were taken into account during the development of DIAS. Thus, our developed tool was decided to also provide developers the capability to understand how various disabilities would affect HCI with their products, even in the cases where the impaired end user utilizes the functionality offered from common assistive technologies, namely magnifiers and screen readers.

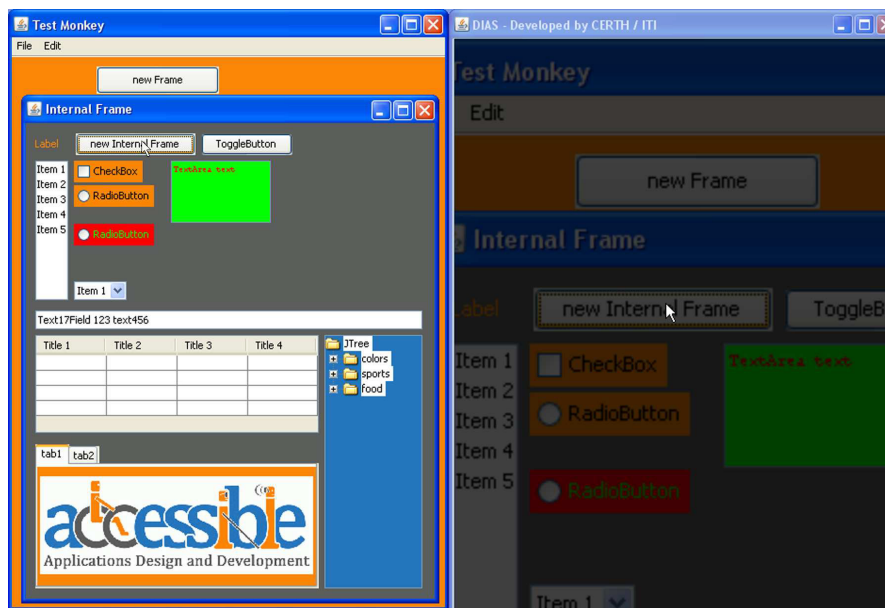


Figure 119: Magnifier applied over night blindness simulation

In order to simulate the result of magnifiers applied over the UI under examination, DIAS utilizes image resizing capabilities offered from the openCV library. In particular, the filtered UI is resized so as to obtain two, three or four times its initial size. Then, the aforescribed disability simulation filters are applied over the portion of the resized UI, which matches the initial user's visual field, and is centred to the position that the mouse pointer has during the simulated interaction. Figure 119 presents the result of night blindness simulation, applied on the portion of the UI that is presented to the user, after "3x" magnification.

Further to magnification, the simulation of visual impairments in the presence of screen readers is also realized with our tool. This is done through the integration of the NVDA screen reader within the simulation environment. It should be noted at this point that for the simulation of how Java Swing applications are perceived by the means of NVDA, DIAS uses also the Java Access Bridge, which allows the screen reader to obtain information regarding the components that comprise the application under examination. Similarly to the screen magnifier case, the user of the tool is capable to view the simulation of any supported impairment, accompanied by a screen reader that simultaneously provides information in the form of sound, relative to the UI elements that are focused or manipulated during interaction.

As a result, the developer of a UI can first of all understand how a totally blind person will perceive the interface. This way s/he can examine for instance, whether the focus order of the UI components the application consists of is appropriate, and also, whether appropriate assistive information is provided to the screen reader through the application, so as for the latter to be eventually properly presented to a user that is totally blind. In order to do so, the developer can simply select to simulate the application in respect of the “total blindness” disability, supported by the tool. In this case, the NVDA is automatically started, and set to operate over the application under examination, as the user interacts with it. Once the user selects a different disability to simulate, NVDA is automatically shut down.

Moreover, it should be noted that screen readers may as well be used from people with visual impairments, who although not completely blind, have very limited visual perception. DIAS can also provide the developer with insight over such HCI cases.

This is done by providing the user with the capability to use the screen reader also while examining the given application over any of the supported impairments. In this case, while simulating an impairment, the user may enable or disable the integrated screen reader at any time, by simply clicking over the respective option. Of course, both the magnifier and the screen reader can be used simultaneously, also in conjunction with any supported impairment.

The screen reader functionality integrated in DIAS can be particularly helpful in several cases of severe visual impairments, such as severe glaucoma etc. and in both cases of Java-based and web-based UIs. For the impact of the incorporated screen reader functionality to become clear, imagine the case of a web page, which requires user input within text fields, however, the text fields of the UI are not appropriately connected within the HTML code to the labels preceding (and explaining) them. Using only the visual simulation of a user with severe glaucoma, the developer can easily identify that her/his web page will not very easily utilized by such a potential user. Nevertheless, s/he could consider that a screen reader would be helpful, allowing the impaired user to easily understand what kind of input should be provided. By utilizing the screen reader capability integrated in the DIAS tool, the developer can further examine the web page and understand from this simulation that these “missing links” between the required user input and their description (preceding labels) lead to a result that is hardly accessible to a user with severe glaucoma and also inaccessible to a totally blind end user. This way, the developer becomes instantly aware of accessibility problems that this web page has, having thus the capability to subsequently fix the errors identified through simulation.

7.5. Disability Impairment Approximation Simulation (standalone version)

The main view of the tool is depicted in the following Figure 120. The user has two options. In the first option he/she can either simulate a Java Swing Application by pressing the corresponding button.

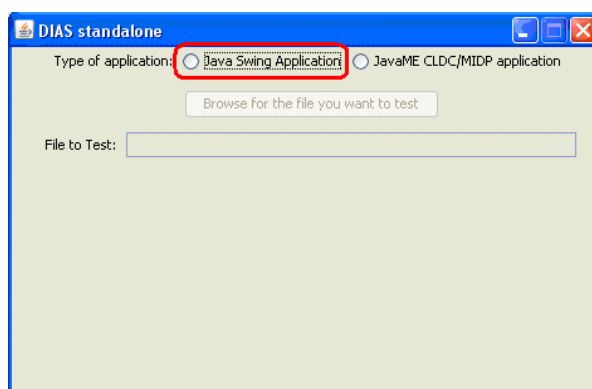


Figure 120: DIAS standalone, main view

In order for a Java application to be simulated, the user of the tool should simply browse for the .jar file of the application and select it. Then, through the “Start DIAS” option, the application is automatically started and presented within the DIAS simulation environment.

Figure 121 presents an instance of this environment during the simulation of glaucoma. The simulation environment consists of two panes: the interaction and the

simulation ones. The interaction pane presents the actual application, in its original form, with which the tool user can freely interact.

The simulation pane presents how the same UI would have been perceived from impaired end users. Within this pane, the user can select any of the supported impairments or incorporated personas, through the tree structure marked in red in Fig. 8. For each selected disability, the user can select a specific intensity level to be simulated, through the slider marked in Figure 121 in blue. Below this slider, statistical information regarding populations suffering from the specific symptoms are provided, based on the relevant literature, so as to further facilitate the developer towards understanding the impact of the problems identified through simulation.

Apart from the pre-defined severity levels, the user can also set for each impairment specific values regarding the parameters involved in its simulation. This can be done through the “Show Detailed Settings” option, shown in Figure 121. In this case, the severity level selection slider is replaced by ones that regard more specific parameters affecting the filters involved within the given impairment. Figure 122 shows the result of such a replacement, in respect of the Parkinson’s disease. Herein the user can set different values for the amplitude and frequency that the simulated tremors should have. It should be noted at this point that through the options provided next to the “Detailed Settings” one (Figure 121), the user can also enable the afore-described Magnifier and Screen Reader functionalities, within each simulation.

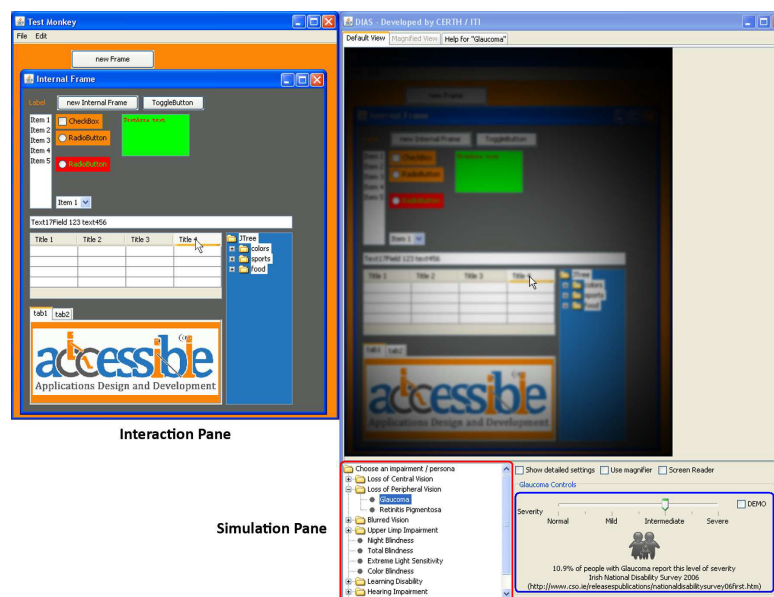


Figure 121: The DIAS simulation environment

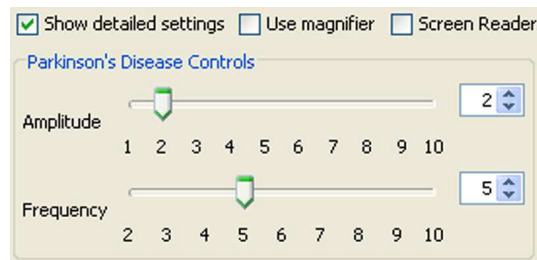


Figure 122: Detailed Settings for the simulation of parkinson's disease

As shown in Figure 121, a Help tab is also provided within the simulation pane of DIAS, in respect of the supported disabilities. The Help tab provides additional information about the simulated disability, thus further facilitating the user to understand its symptoms. In Figure 123, the Help tab is depicted, concerning the simulation of Tritanopia.

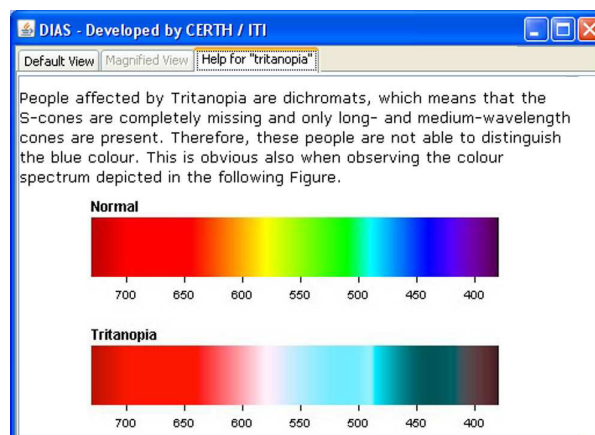


Figure 123: Help tab for Tritanopia

The user can also use the *Magnifier* to each simulation. This feature was added, because, based on the results of User validation of DIAS, almost all people who suffer from a visual impairment, use the feature of magnifier when using a computer. In Figure 124 a simulation of Achromatopsia is depicted, using also the Magnifier.

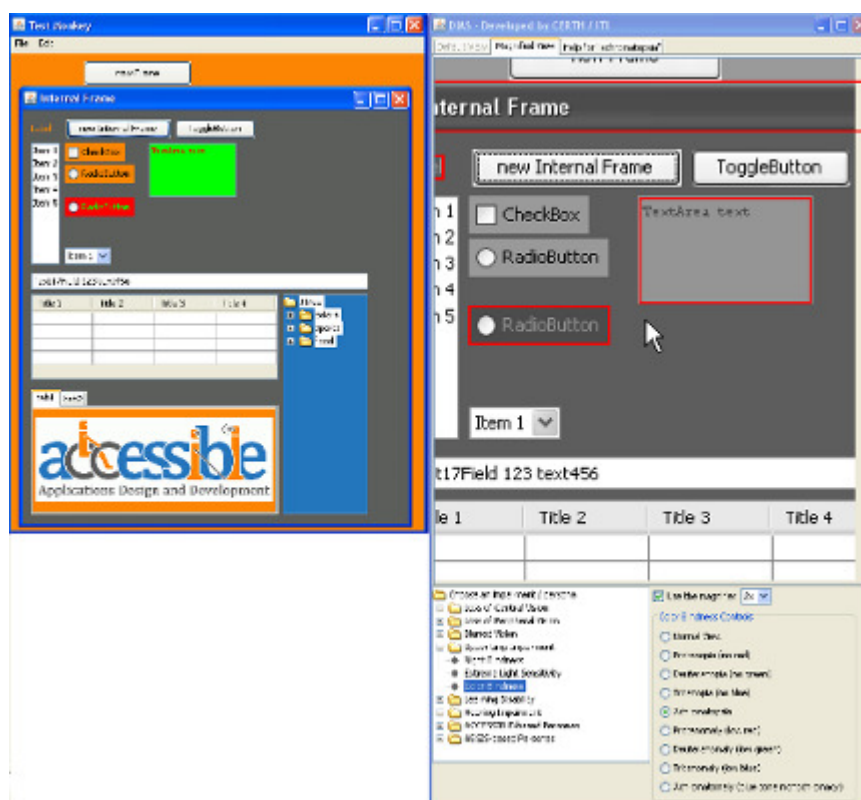


Figure 124: Simulation of Achromatopsia using Magnifier

The second option for the user is to choose to simulate a JavaME CLDC/MIDP application. Thus, as it can be seen in Figure 125 the user is offered with two options, with the first one being the already implemented Java Swing functionality and the second one, being a new functionality for the simulation of Java mobile applications. Instead of a .jar file, the user chooses a .jad file, he/she then chooses the emulator platform, then a specific mobile device and the simulation process is the same as previous. The emulator platform that was used is the Sprint emulator [5] and in Figure 126 a simulation of protanopia is depicted.

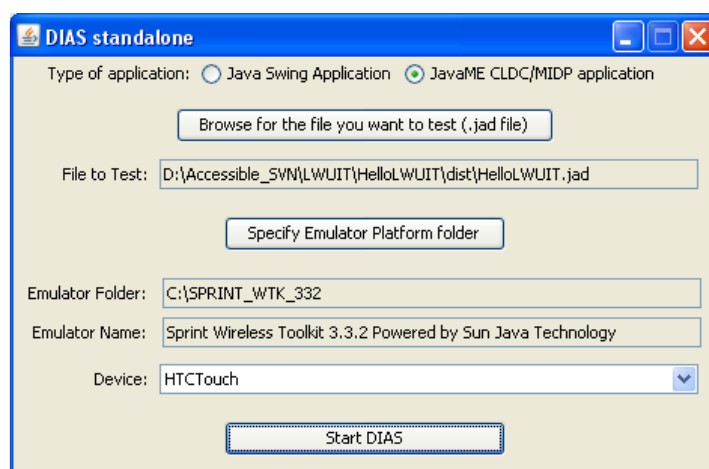


Figure 125: DIAS standalone, initial state

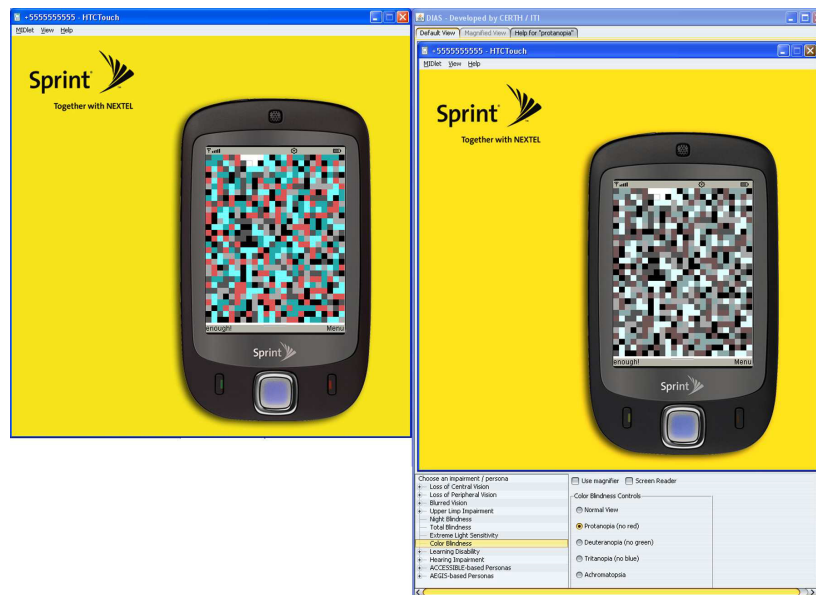


Figure 126: Simulation of protanopia over a Java-based mobile application with DIAS

7.6. Disability Impairment Approximation Simulation (NetBeans plugins)

A suite of three different plugins for the NetBeans platform has been developed, in order to provide users of this IDE with the DIAS impairment simulation capabilities, embedded in their development environment. As explained in the following, two out of these three plugins enable the simulation of Java applications, whereas the third one allows the simulation of web pages of sites that are developed through the NetBeans IDE.

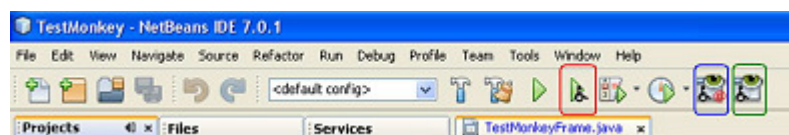


Figure 127: Toolbar of NetBeans IDE with the DIAS run main (marked in red), web preview (blue) and preview design (green) plugins installed

Java applications

In respect of Java applications, the DIAS NetBeans plugin suite includes the preview design and the run main simulation functionalities. The preview design plugin provides users a visual design preview feature that allows them to understand how their implemented Swing forms can be perceived from people with visual impairments. This plugin enables developers to preview the form layout of their implementations, similarly to the simple “preview design” of the NetBeans platform, yet along with impairment simulation. The preview design functionality can be activated by clicking on the appropriate small image of an eye with an arm-chair preview icon, at the IDE’s Toolbar (Figure 127), while the developer is viewing a Swing form within the NetBeans conventional GUI form designer. Figure 128 shows an instance of the DIAS preview design plugin, while glaucoma is simulated over a Swing form that is being developed with NetBeans.

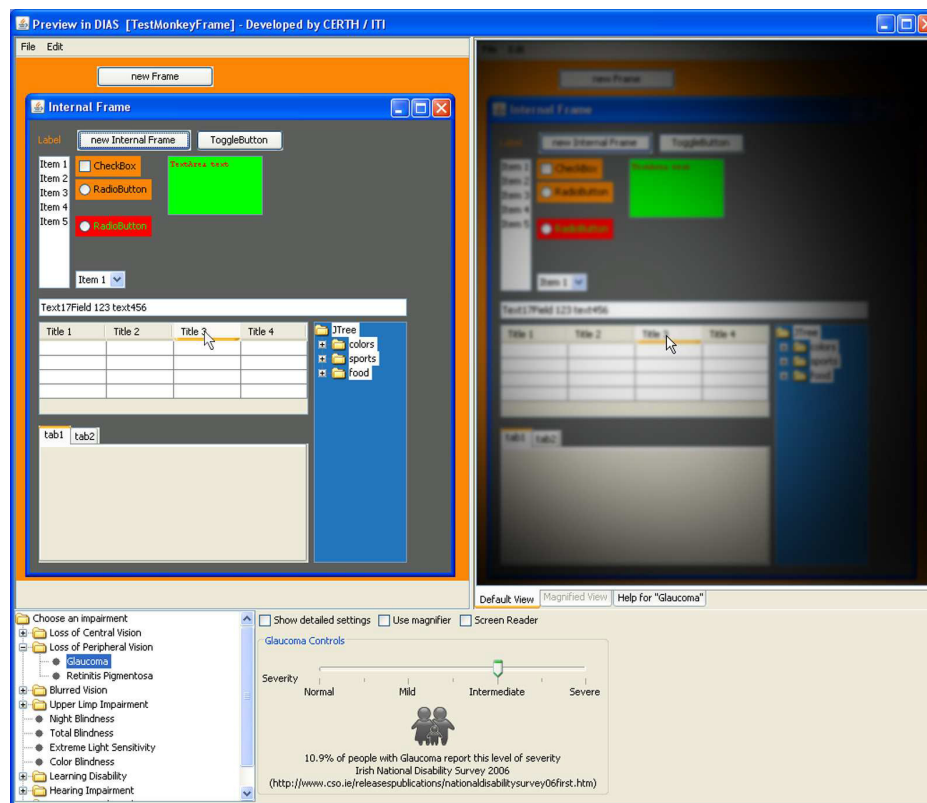


Figure 128: Simulation of glaucoma over a Swing form through the DIAS preview design plugin

Apart from the ability to see an approximation simulation of various disability impairments the developer is presented with a list of all the accessibility errors and warnings regarding the implemented GUI form. Thus, s/he can inspect the problematic GUI components and get a short description of the problem and/or a recommendation on how to fix it. When a specific error or warning is selected, the respective GUI component is highlighted in the DIAS simulation pane. Additionally, the properties regarding the component are presented, as it is shown in Figure 129. When a warning / error is selected, the developer can double click on the recommendation window in order to fix the identified problem.

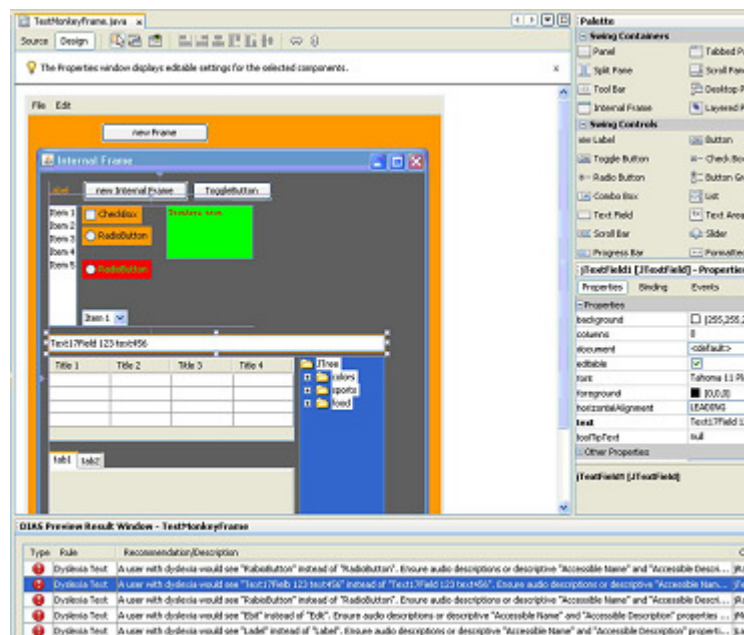


Figure 129: Recommendations provided through the NetBeans IDE preview design plugin, in respect of dyslexia simulation

The preview design plugin can facilitate a developer with limited knowledge on accessibility issues towards building accessible Java Swing forms. However, the real-time interactions of developers with their Swing application forms are also highly important within the design of their implementations. Static and non interactive views of applications are fine when developers and designers are developing independent forms, but the preview design plugin cannot assist them in respect of understanding how form components should behave as the application is running in real time, where relevant modifications come up during interaction (e.g. modification of size, usage of combo boxes and buttons, change the ordering of tabs, etc). To overcome this issue, the DIAS run main NetBeans plugin has been implemented, so as to enable developers to explore, run and test their implementations during simulation of impairments. While their applications are running, new windows, such as dialogs, choosers or frames, may appear due to user interactions. The DIAS run main functionality can be activated by clicking on the appropriate image of a green triangle with an arm-chair, at the IDE's Toolbar.

(Hyperopia) is simulated over a Swing form that is being developed with NetBeans. As shown in Figure 130, the DIAS UI presented to the user is in this case the same as the one presented during Java application simulation through the stand-alone version of the tool.

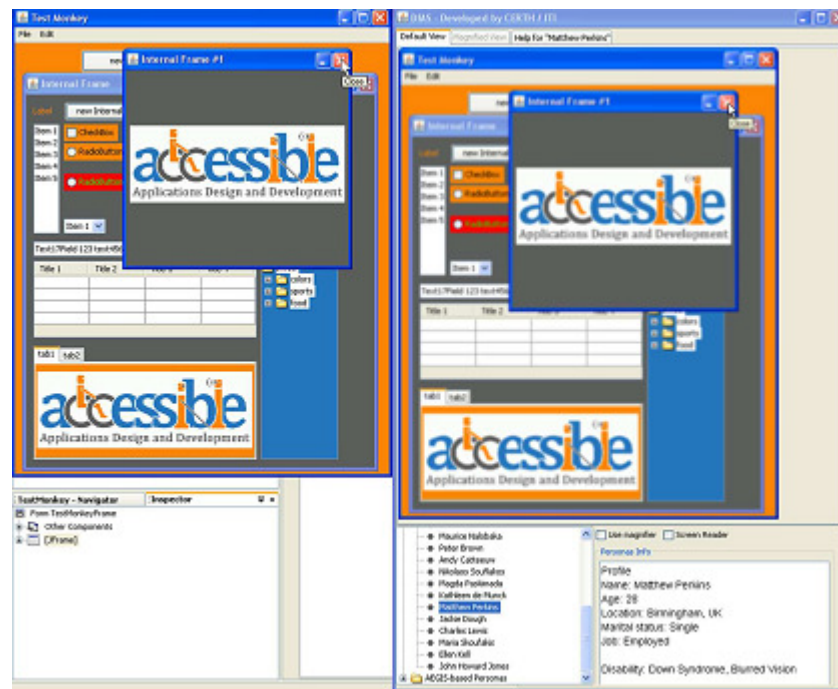


Figure 130: Simulation of a Java application with a persona suffering from blurred vision (Hyperopia) through the DIAS run main plugin

Web applications

NetBeans provides developers with the capability to create web sites, consisting of both static (e.g. .html) and dynamic (e.g. .jsp) web pages. Our developed tool offers developers using NetBeans also the capability to examine the accessibility of web-based UIs. This is achieved through the web preview plugin of the DIAS NetBeans suite. Typically, web pages developed through NetBeans can be viewed through a “Preview” option offered by an IDE’s plugin. Our plugin introduces in NetBeans a further option, through which web pages can be viewed, through simulation of our tool’s supported impairments.

Within the DIAS web preview plugin, the XULrunner runtime is utilized through the DJNativeSwing library, so as to parse web pages and present them. As shown in Figure 131, this plugin shows the web page, as provided from XULrunner on the left pane, where the developer may interact with it, similarly as a user would do so using a XUL-based web browser (e.g. Mozilla Firefox). Simultaneously, the result of the web page, filtered through the DIAS impairment filters, is presented at the simulation pane of the plugin’s UI, similarly to the afore-described preview design and run main plugins of Java applications. As a result, the developer can gain an insight of how her/his developed web page would be perceived from users with the supported disabilities.

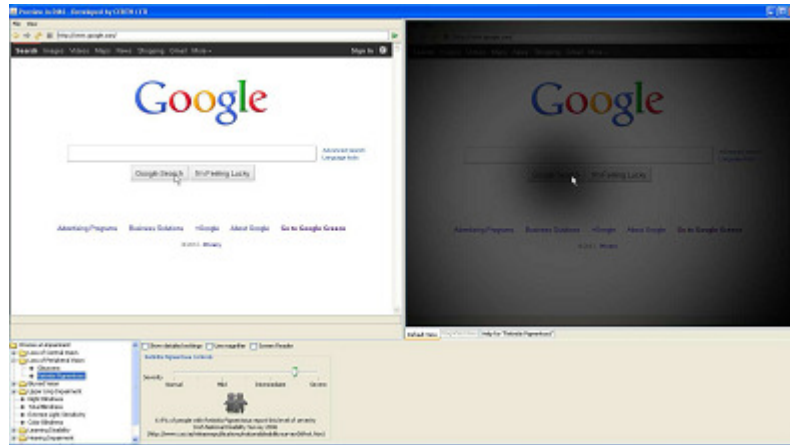


Figure 131: Simulation of a web page (www.google.com) with retinitis pigmentosa impairment through the DIAS web preview plugin

7.7. Changes in current version

A number of changes were made concerning both version of DIAS. Some of them apply to both versions, but others apply to one of the versions. Thus, since the last issued version of the tool a number of changes were made that are summarised in the following:

- The simulation filters for most of the visual impairments such as cataract, protanopia, etc. were improved in order to more accurately describe the corresponding limitations.
- All the bugs found in the pilot tests (phases 2 and 3) were corrected
- The description of the provided recommendations in the plugin version of the tool was modified in order to more accurately point out the required modifications towards correcting accessibility limitations.
- The functionalities of the DIAS (Netbeans plugins) were further improved in order to offer developers using NetBeans the capability to examine the accessibility of web-based UIs. This is achieved through the web preview plugin of the DIAS NetBeans suite. Thus the XULrunner runtime was utilized through the DJNativeSwing library, so as to parse web pages and present them to the user.

8. Introduction to Mobile Impairment Simulator (MIS Tool)

Mobile Impairment Simulator (MIS tool) is a standalone desktop simulation application for mobile application developers. It can provide simulation of common visual impairments as well as issues from indoor and outdoor physical environment. These simulations can be mixed in various combinations.

Motivation for the MIS tool

Importance of mobile applications is growing rapidly as it provides access to variety of very useful services. Moreover the penetration of mobile phones into everyday life is higher than any other ICT. Especially a smartphone became a useful assistant for everyday life. In particular the impaired users found it a very powerful supportive device in various situations. It is possible to say that every impaired ICT user is using at least mobile phone. Therefore it is necessary to keep in mind the accessibility of mobile applications in order to avoid exclusion of impaired users. Unfortunately the field of mobile applications and devices lacks sufficient effort towards their accessibility. Ignoring the accessibility of mobile applications and devices affects in the largest extent users with visual impairment who create a relatively large segment of the market. Therefore we focus primarily on this user group.

In comparison to desktop computers or laptops that are usually used indoor (at home, in the office or in a restaurant), mobile devices are often used in different physical environments like in public transport, on the street, in the park, etc. We have analyzed typical issues that can occur in these environments.

Issues specific for mobile world

One of common problems is the low contrast of the user interface on the display. This situation is more painful especially when the display is on direct sunlight or the display reflects the surrounding landscape, or face of the user like in the mirror (See left side of Figure 132). The other problem with UI visibility is caused by changing reflections while moving, for example changing reflections of nice countryside when travelling in the bus or train. Finally if we take into account critical combination of environmental conditions and particular visual impairment we can identify other problematic situations. For example if there occurs a sharp change in the lighting conditions (like leaving a tunnel, strong source of light appearing accidentally behind the display) and the user suffers from light sensitivity impairment, it can take much more time till the eyes will accommodate to the new situation. This can lead to serious accessibility issues (like missing an important alert, impossibility to react in a given response time).

The usage of the mobile devices on the move brings several other issues such as display tremor which makes especially small non-contrast text hard to read.



Figure 132: Reflection on the display and occlusion of the display with finger

Another set of specific issues is related to the user interaction with the mobile device. Touch screen devices bring another issue in a form of occlusion of the display by interacting fingers (see right side of Figure 132). Information we are looking for is displayed under the finger and we simply do not see it. This occlusion can be even bigger if the user suffers from visual impairment like low-vision on one eye and blindness on the other one.

One example of necessity to focus on good accessibility could be filling of the large forms on small displays especially when half of the display could be occupied by software keyboard. In cases where part of the display is covered when filling form there is necessity of good traversing order between items of the form. We have to ensure not losing the context and provide information which item is currently displayed and filled. This problem often occurs when layout of display is changed from portrait to landscape mode and vice versa. Other use case of mobile accessibility issue can be found in [9].

All usual accessibility guidelines focus strictly on applications themselves and do not take into account physical environment that also influences the accessibility in mobile environment. There are also tools helping with accessibility of web pages for mobile devices like the W3C MobileOK Checker [10]

Issues that should be simulated

Some situations can be easily simulated by the developer but some not. For example the issues related to small screen can be evoked by usage of emulators or deployment of application to the target device. The dynamically changing reflections on the display, trembled display caused by ride on the bus is hard to simulate with the common tools. The changes of environmental light can affect also the ability of the visually impaired users to read the display. Simulation of different light sources and their intensity, dynamic changes and special situations (like moving reflections) cannot be easily simulated in the developer's office. Occlusion of the display caused by the finger touching the display while having visual impairment is very hard to simulate without any special simulator.

We have focused on these hard to simulate situations and developed a simulator which can easily provide an illusion of these situations to the developers.

Simulation tool architecture

In comparison to the world of desktop and web applications there are many mobile platforms that are rapidly emerging and disappearing on the market and all the work related to accessibility can be easily lost with the discontinued mobile platform.

Therefore it is essential to develop simulation tools which are as independent as possible on the mobile platform emulators and integrated developer environments.

We have developed a simulator - Mobile Impairment Simulation tool [8] – which is implemented as a standalone application. It is totally independent on the mobile platform for which the mobile application is being developed. The simulation is done by means of on-the-fly manipulation with the pixels rendered on the screen typically by some mobile platform emulator.

The simulator consists of 2 main parts - Filter settings window and Simulation window:

- **Filter settings window** (see Figure 133) where effects and their parameters can be adjusted. At the bottom of the Filter settings window is placed the integrated context help. All three sections of filter settings window are described below.
 - Section with adjustable control parameters of currently focused simulation filter (see Figure 133 section 1)
 - Stack of simulation effects that are currently applied in Simulation window. These effects are automatically grouped and sorted into three logical levels (Eye, Middle and Display) according to real world order. (see Figure 133 section 2) Thus the correct order of applied filters is always ensured.

Integrated context helps explaining the use and control parameters of particular filters.
(see Figure 133 section 3)

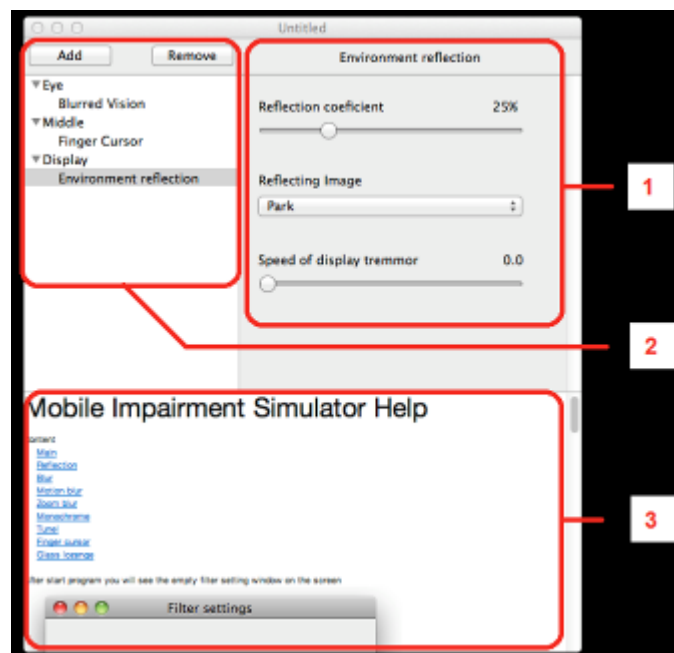


Figure 133: Screenshot of Filter settings window with marked sections

- **Simulation window** is place where simulation effects take place. On the image (see Figure 134) is applied effect of reflection from fluorescent lamp on the display, finger occlusion of the display and the effect of blurred vision.

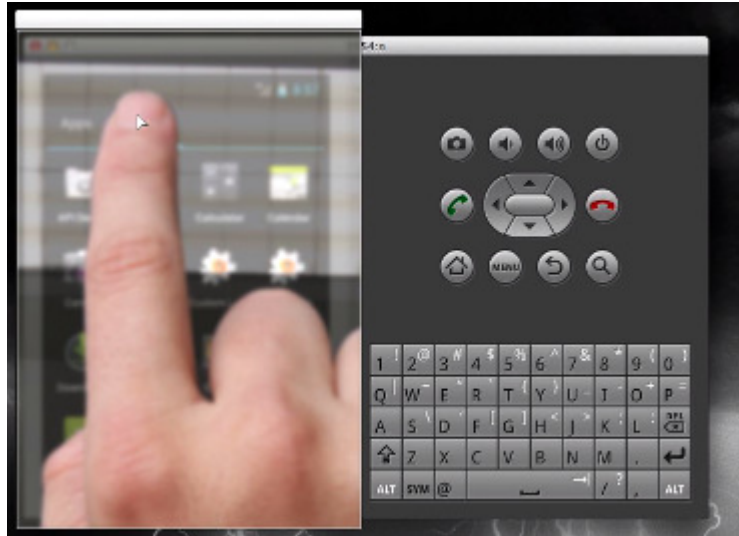


Figure 134: Screenshot of MIS tool with blur effect placed over the Android emulator

Currently there are implemented following simulation effects:

- Physical environment effects
 - Static reflection
 - Dynamic reflection
 - Display tremor
 - Finger occlusion
- Visual impairments effects
 - Tunnel vision
 - Blurred vision
 - Colour blindness
 - Macular degeneration
 - Extreme light sensitivity

It is also possible to combine parameters and create real life situations such as reflection on the display and finger occlusion or display tremor and blurred vision. The static reflection effect provides preview of the mobile application when for example fluorescent lamps (see left side of Figure 135) on the ceiling reflects on the display. Developer can easily adjust strength of the static reflection effect and check if the user interface is still readable. Finger occlusion effect is implemented by modified mouse pointer that is attached to the photo of index finger or thumb (see right side of Figure 135). Usage of mobile device while walking or travelling by car or by bus brings tremor of the display and introduces text reading problem. Display tremor effect is simulated by random motion blur effect.

Many of these effects can be also used to check the usability of the visual design for users with no visual impairment.



Figure 135: Screenshot of simulated effects - reflection of fluorescent lamp and finger occlusion

Implementation of MIS tool

Mobile Impairment Simulator is developed for Mac OS X and uses Cocoa framework for rendering of effects. Most of the filters are implemented using Core Image, Apple image processing technology that leverages programmable graphics hardware whenever possible to provide near real-time processing. Core Image API provides access to built-in image filters for both video and still images and provides support for creating custom filters. Unfortunately this technology is very closely tied to Mac OS X environment and there is currently no similar multi platform solution.

Limits of used approach

As it was mentioned previously, Mobile Impairment Simulator is fully independent on any type of mobile platform emulators. There is no relation between the mobile platform emulator and Mobile Impairment Simulator. Static image rendered by mobile platform emulator is periodically sent to Mobile Impairment Simulator, where it is modified according to requested simulation effect. Therefore we do not know the semantics of the objects on the display (location of buttons, labels, input fields, etc.) thus issues like wrong tab-traversal cannot be simulated. This is the most important limit of the approach we have used for simulation. We are considering the fact that some kind of integration can be useful as it can make the development process smoother and faster. Therefore it is possible to start MIS tool from command line with set of parameters including the XML file where setup of particular simulation effects can be predefined.

A useful extension can be an implementation of plugin for developer's IDE which will allow automatic start and comfortable parameterization of Mobile Impairment Simulator.

Pilot testing and changes performed in the MIS tool

MIS tool has been tested in pilot tests with developers and several important usability issues occurred. In this chapter we describe the most important issues and changes that have been performed.

In prototype of MIS tool each filter settings window was opened in new window. This lead in too many opened floating windows and resulting to bad and confusing window arrangement. Filter settings windows have been redesigned and now there is just one

window for all filters. Particular filters that are applied are displayed in the Stack of simulation effects (see section 2 of Figure 133).

Combination of multiple filters had to be reordered manually and could lead to wrong order of filters resulting in unrealistic simulation. We have implemented partially automatic ordering of filters. Each filter now belongs to one of three predefined categories of layers (Eye, Middle, and Display) (see section 2 of Figure 133). Thus correct order of filters is ensured.

Simulation window did not support scaling, thus landscape mode applications or tablet applications could not be simulated. Therefore simulation window currently supports free scaling and so it can be adjusted according to the needs of developer.

Position of the simulation window was not remembered and has to be adjusted each time. Now the last position of the simulation window is saved automatically and used each time the MIS tool is started again.

After starting of MIS tool, there was no window displayed what was very confusing for the users. Now after starting of the MIS tool the filter settings window appears immediately to provide necessary feedback to the users.

Parameters of the filters were not clearly labelled and explained. Therefore we have revised and renamed all control parameters of filters to provide clear user controls. Some filters have been also renamed (see section 1 of Figure 133).

We have also implemented support for loading XML configuration files where all the filters and its parameter values could be pre-defined. Thanks to this feature it is not necessary to adjust all frequently used setups manually each time. It is for example possible to create simulation use cases like simulation of personas or real life situations (using mobile device while walking in the street during the sunny day).

9. Conclusions

This deliverable (software and report) outlines the description of the final implemented ACCESSIBLE prototypes, which include the Web Accessibility assessment module (integrating also the ARIA assessment module), the Web Services Accessibility assessment module (supporting also the WADL services), the Mobile Web applications assessment module, and the Description Languages assessment tool which has been integrated in the SAFIRE IDE.

The detailed description of the technologies that were used in order to implement each module, along with the general architecture and the system requirements, were presented in Deliverables D2.2b, D3.3b, D5.2, D5.3 and D5.4. For each module, apart from the changes that were made since the last version (according to the pilot phase 2 and 3), a detailed description of the tools functionalities is being provided, in order to better clarify the way a user can interact with each module.

10. Annex

10.1.ACCESSIBLE Ideal Operations of the Web Services Assessment module

An “Operation Alignment” procedure is incorporated in the ACCESSIBLE WS Accessibility Assessment procedure, which aims to align an Operation (its inputs and outputs) defined within the Web Service under assessment to one of some “Ideal Operations” defined within the ACCESSIBLE WS Accessibility Assessment Tool. The purpose of the Operation Alignment process is for the Tool to understand the “type” of the WS Operation under assessment, since some WS Accessibility Guidelines are applicable to specific types of Operations and some of the WS Guidelines can be evaluated only after the user provides specific alignment information regarding her/his Operation’s outputs and/or inputs. For example, the “type” of the operation can be “Image Provider” in case that the operation provides image(s) upon invocation, “Textual Info Provider” in case it provides textual information as output, etc.

“Operation Alignment” is thus the process during which an Operation of a WS is declared to be of a specific type, by aligning it to one of the “Ideal Operations” defined within the Tool. During this process, the user selects an “Ideal Operation” that s/he believes is more similar to the Operation under evaluation, and then the inputs and outputs of the Operation are semi-automatically aligned to corresponding ones that belong to the selected “Ideal Operation”. For example, if the operation that the user wants to assess for accessibility provides images as output, then the user shall align it to the ACCESSIBLE “Image Provider” Ideal Operation.

In the following, an overview of the ACCESSIBLE Ideal Operations, defined within the ACCESSIBLE WS Assessment tool is provided:

General Services

1. Image Provider Operation

Description: An operation that provides one or more images as output

-Input

NONE

-Output

1. Accessible Image
 - i. Image Object
 - ii. Image Object URL
 - iii. Alternative Text

2. Audio Provider Operation

Description: An operation that provides one or more audio files as output

-Input

NONE

-Output

1. Audio Object
2. Audio Object URL

3. Title
4. Short Description
5. Full Transcript\

3. Video Provider Operation

Description: An operation that provides one or more video files as output

-Input

NONE

-Output

1. Accessible Video
 - i. Video Object
 - ii. Video Object URL
 - iii. Title
 - iv. Short Description
 - v. Full Transcript
 - vi. Audio Description
 - a. Audio Description Object
 - b. Audio Description Object URL
 - c. Audio Description Title
 - d. Audio Description Short Description
 - e. Audio Description Full Transcript
 - f. Audio Description Sign Language Interpretation

4. Textual Info Provider

Description: An operation that receives input or provides as output information in the form of text

-Input

1. Desired Language

-Output

1. Accessible Text
 - i. Text Object
 - ii. Text Object URL
 - iii. Image Of Text
 - iv. Text Language
 - v. Textual Info In English
 - vi. Link To Auditory Or Graphic Presentation
 - vii. Simple Text for Secondary Education Level

Info-mobility Services

1. “Points of Interest” Info Provider Operation

Description: An operation that provides information about “Points of Interest” as output

-Input

1. User Category

-Output

2. Accessible Point of Interest
 - a. Point of Interest
 - i. User Group
 - ii. POI Accessibility Status

iii. POI Accessibility Status Details

2. “Route calculation” Operation

Description: An operation that provides route calculation capabilities and produces “calculated routes” as output

-Input

1. User Category

-Output

1. Accessible Route
 - a. Pedestrian Route Segment
 - i. User group
 - ii. Accessibility Status (C2.1 – L3m)
 - iii. Accessibility Status details (C2.2 – L3)
 - b. Public Transport Route Segment
 - i. Origin Accessibility Status
 - a. User group
 - b. Accessibility Status (C2.3.1 – L3m)
 - c. Accessibility Status details (C2.4.1 – L3)
 - ii. Destination Accessibility Status
 - a. User group
 - b. Accessibility Status (C2.3.2 – L3m)
 - c. Accessibility Status details (C2.4.2 – L3)
 - iii. Means (Vehicle)
 - a. User Group
 - b. Accessibility Status (C2.3.3 – L3m)

Accessibility Status details (C2.4.3 – L3)

10.2.Controls for the Supported Impairments in DIAS

For each supported impairment different factors can be controlled through the controlling toolbar of the DIAS tool. By these means, each impairment can be approximately simulated for different degrees and levels of severity. The controls that were implemented for DIAS tool are influenced by the Visual Impairment Simulator for Microsoft Windows³, a tool which can simulate various visual impairments. In Table 7 a short description for each control is presented, while in Table 8 the usage of the simulation controls is depicted.

<i>Simulation Control</i>	<i>Description</i>
Blur Strength	The strength with which the area affected by the impairment is blurred, simulating general loss of visual acuity
Yellowing	The degree to which the area of the cataract is made yellowish-brown,

³ <http://vis.cita.uiuc.edu/>

	simulating the yellowing phenomenon that appears in some cases of cataracts
Disc Size	The size of the affected area
Disc Count	Controls the number of discs that are used to simulate the impairment. If this value is high, the disc will be more irregular, and if it is low, the impaired area will appear mostly or completely circular
Disc Spread	Controls how spread out the discs are that simulate macular degeneration. If the setting is high, the impairment will appear as a much wavier disc or even separated discs. If the setting is low, the impairment will appear mostly circular
Relative Position to Mouse Cursor	This setting was created to simulate the fact that many people with macular degeneration look out of the corner of their eye. So, the position of the centre of the impairment can be moved slightly, simulating this effect
Severity Level	Controls both the size of the unimpaired disc and the darkness of the impaired area. As the severity increases, the amount of unimpaired vision gets smaller and the impaired vision gets darker
Tunnel Size	The size of the area that is not yet completely black
Whitening	This simulates various degrees of a whitening effect where the entire inside of the visual field loses acuity
Transition Width	This setting changes how quickly the vision falls off to black. If it is set low, the transition will be a fairly sharp fade to black, if it is set to high, the transition will be gradual, starting at the centre of the visual field
Brightness	Controls how strongly the vision is illuminated
Amplitude	Controls the amplitude of the Parkinson's disease tremor

Frequency	Controls the frequency of the Parkinson's disease tremor
Volume	Controls the volume of the system
Sensorineural Hearing Loss Controls	This setting controls the different types of sensorineural hearing loss that can be encountered
Dyslexia Controls	This setting controls the different symptoms of dyslexia that can be encountered

Table 7: Description of the simulation controls

Impairments Simulation Controls	Cataract	Macular Degeneration	Glaucoma	Retinis Pigmentosa	Hyperopia	Night Blindness	Extreme Light Sensitivity	Parkinson's Disease	Conductive Hearing Loss	Sensorineural Hearing Loss	Dyslexia
Blur Strength	x	x	x		x						
Yellowing	x										
Disc Size	x	x									
Disc Count		x									
Disc Spread		x									
Relative Position to Mouse Cursor		x									
Severity Level			x								
Tunnel Size				x							
Whitening				x							
Transition Width				x							
Brightness						x	x				
Amplitude								x			
Frequency								x			
Volume									x		
Sensorineural Hearing Loss Controls										x	
Dyslexia Controls											x

Table 8: Usage of simulation controls

References

- [1] Evaluation and Report Language (EARL) 1.0 Schema –at <http://www.w3.org/TR/EARL10-Schema/>
- [2] jsoup: Java HTML Parser –at <http://jsoup.org/>
- [3] Download jsoup –at <http://jsoup.org/download>
- [4] Accessible Rich Internet Applications (WAI-ARIA) 1.0 –at <http://www.w3.org/TR/wai-aria/>
- [5] Sprint emulator –at http://developer.sprint.com/site/global/home/p_home.jsp
- [6] National Disability Survey 2006 –at http://www.cso.ie/releasespublications/documents/other_releases/nationaldisability/National%20Disability%20Survey%202006%20First%20Results%20full%20report.pdf
- [7] Koch, J., Velasco, C.A., Ackermann, P. (eds.) (2011). HTTP Vocabulary in RDF 1.0, W3C Working Draft 10 May 2011, <http://www.w3.org/TR/HTTP-in-RDF10/>.
- [8] Mobile Impairment Simulation tool
<http://cent.felk.cvut.cz/hci/accessible/www/index-mis.htm>
- [9] Vystřil, J. , Mikovec, Z. (2011) Accessibility issues simulation for mobile application
https://cent.felk.cvut.cz/hci/accessible/downloads/ACCESSIBILITY_ISSUES_SIMULATION_FOR_MOBILE_APPLICATIONS.pdf
- [10] Mobile OK W3C checker
<http://validator.w3.org/mobile/>